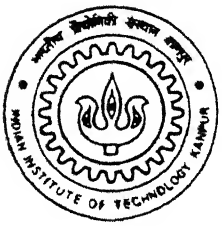
SIMULATION AND IMPLEMENTATION OF A FUZZY VECTOR CONTROLLED INDUCTION MOTOR DRIVE SYSTEM

by

ANANT TEWARI



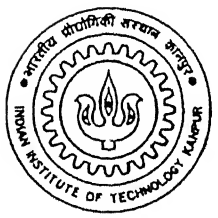
DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

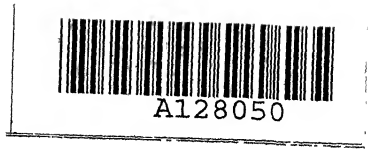
April, 1999

SIMULATION AND IMPLEMENTATION OF A FUZZY VECTOR CONTROLLED INDUCTION MOTOR DRIVE SYSTEM

a thesis submitted
in partial fulfilment of the requirements
for the degree of
MASTER OF TECHNOLOGY
by
ANANT TEWARI



to the
DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
APRIL, 1999



CERTIFICATE

This is to certify that the work contained in this thesis entitled ***Simulation and implementation of a fuzzy vector controlled induction motor drive system*** by **Anant Tewari** has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

SPDas 21.4.99

Dr. Shyama Prasad Das
Department of Electrical Engineering,
Indian Institute of Technology, Kanpur

ABSTRACT

In this thesis, an induction motor drive incorporating a fuzzy logic controller to control its torque indirectly has been simulated and practically implemented .The present work exploits the advantages offered by a fuzzy logic controller (intuitiveness, simplicity, easy implementation, and minimal knowledge of system behavior) to obtain a robust, fast, and precise closed-loop control of speed of induction motor. The control algorithm of the drive system has been implemented by a pentium PC, which uses PCL-208 data acquisition card for communicating with the inverter and the motor. The simulated and practical results show that fuzzy logic controller can give fast torque response and precise control of speed of induction motor drive system.

Keywords: *Indirect field oriented control, Vector control, Fuzzy control, Data acquisition card*

Dedicated
To
My
Parents

Acknowledgement

I wish to express my deep and sincere thanks to Dr. Shyama Prasad Das, my project guide, for his invaluable suggestions and guidance to venture into the arena of vector control of three-phase induction motors. His constructive criticism went a long way in completing and improving my M.Tech thesis.

I would also like to thank my senior colleagues Mr. Ramesh K. Tripathi, Ms. Malabika Basu, batch mates Mr. Anshuman Tripathi, Mr. Anil Bisht, Mr. Anuj Vyas and Mr. Manmohan Singh, whose association made my stay at IIT Kanpur a memorable one.

My thanks are also due to Mr. O. P. Arora, incharge P.E. lab., for his timely technical advice rendered. I am again thankful to all those who have directly or indirectly helped me in the completion of the thesis work.

I shall be failing in my duty if I do not thank to my family members specially my sister Usha, brother-in-law Mr. V. C. Mishra and brother Dinesh, who have been a constant source of encouragement for me.

Over and above all, the loving, encouragement and motivation bestowed by parents during my studies is highly appreciable. This thesis is dedicated to them.

Anant Tewari

Table of contents

List of figures

List of symbols

r_s, r_r	Per phase stator and rotor resistance (Ω)
L_{ls}, L_{lr}	Per phase stator and rotor leakage self inductance (H)
L_s, L_r	Per phase stator and rotor self inductances (H)
λ_{ds}, λ_{dr}, λ_{qs}, λ_{qr}	d-q components of stator and rotor fluxes (wb)
M	Mutual magnetizing inductance (H)
L_{sr}	Peak value of per phase inductance between stator and rotor (H)
i_{ds}, i_{dr}, i_{qs}, i_{qr}	d-q components of stator and rotor currents (A)
T_e	Electrical torque (Nm)
w_r	Electrical rotor speed (Rad/s)
w_{sl}	Slip speed (Rad/s)
w_{rs}	Speed reference (Rad/s)
P	Number if poles
i_{as}^*, i_{bs}^*, i_{cs}^*	Three phase stator reference currents (A)
K_s	Transformation matrix for three phase stationary to two phase rotating frame
J	Moment of inertia (Kg-m^2)
B	Coefficient of viscous friction
T_l	Load torque (N.m)
K_p, K_I	PI controller gains

Chapter 1

Introduction

1.1 Background

Alternating current motors are getting more and more popularity for applications in industrial environment due to characteristics of higher efficiency, high torque-to-weight ratio, small volume and lower cost. The capability to operate at higher torque, higher speeds, larger power ratings, robust construction and low maintenance cost, make the induction motor drives more attractive than dc drives.

Earlier dc machine had been the obvious choice for applications in adjustable speed drives where good torque and speed controllability were desired. The drawback with dc machine is that they require frequent maintenance, have low torque-to-weight ratio, higher volume and the sparking at brushes due to mechanical commutation. However, a separately excited dc machine is a doubly excited system leading to independent control over both field flux and armature current, and so by adjusting these parameters flux and torque can be controlled independently. Furthermore, due to low armature inductance the armature current can be changed with good dynamic response for fast torque transients. Since induction motor is a singly excited system, the flux and torque are not inherently decoupled. So their independent control is not possible without a suitable controller. In 1971 Blaschke [1] proposed field oriented control or vector control of induction motor by which induction machine can be controlled quickly and stably by controlling the stator current as a vector quantity with a flux component and a torque component in two ways.

(a) Flux vector feedback method where magnetic flux vector from induction motor is fed back and stator current is controlled on the basis of flux vector (**direct vector control**).

(b) Slip frequency control method [2] where angular velocity of rotor is fed back and is added to calculated slip speed to control the stator current vector (**indirect vector control**).

Field oriented control is the most successful in meeting the requirement of wide speed range and fast torque response. Using field oriented control, a highly coupled induction motor can be controlled through a linear independent decoupled control of torque and flux similar to separately excited fully compensated dc motor.

The basic principle of field-oriented control is to maintain secondary (rotor) flux constant, this also assures the control of flux and torque independently. Several schemes exist to realize field-oriented control [3-7]. In [7], a torque reference is generated with a calculated speed reference and then inverter is switched to track the reference current. The present work uses a fuzzy logic controller, which sets the torque component of current reference based on the speed error and change of speed error. The inverter is switched to follow the set current reference within a hysteresis band, and this control gives a constant rotor flux. Fuzzy logic controller combines human experience as a rules base so it can account for non-linearity in the system behavior into account. Since induction motor also behaves as a non-linear system because of change in resistance (due to temperature change) and change in inductance (due to saturation), a fuzzy logic controller is best suited for controlling an induction motor. Furthermore, the fuzzy rules defining the system behavior can be optimized to get best possible performance.

1.2 Objectives

The present work deals with design, simulation and implementation of an indirect vector controlled induction motor using a fuzzy controller.

1.2 Summary of chapters

The first chapter gives a brief introduction and summary of the thesis. The second chapter is devoted to computer simulation of the proposed system, which contains the equations describing the behavior of induction motor, steps in simulating a fuzzy logic controller and algorithm for generation of computer program. The third chapter deals with the realization of the proposed system and introduces the hardware used, programming needed to generate appropriate signals for control of the drive system. The fourth chapter contains the simulation

and practical results. The conclusion is drawn in the fifth chapter with a scope for future research work.

Chapter 2

Digital Computer Simulation

2.1 Introduction

Before implementing a properly designed system, it often becomes necessary to know the feasibility of proposed system. In practice, behavior of system is investigated through computer simulation. Simulation gives the mathematical solution of equations governing the system performance. The solution of performance equations can predict approximate behavior of the system.

In the present chapter, behavior of the proposed system is investigated by mathematical solution of equations governing system performance. In the end, some simulated results have been shown. In addition, effectiveness of the fuzzy controller for controlling the drive system is demonstrated by comparing system response using the fuzzy controller, with that using a conventional PI controller.

The computer program for simulation of proposed scheme is done in C language. The Runge-Kutta fourth order method has been used for solving the differential equations governing system performance.

2.2 Induction machine model

2.2.1 Machine voltage equations in synchronously rotating reference frame variables

Equation of induction machine in two phase variables [8] can be written as

$$v_{qdos} = r_s i_{qd0s} + w_e \lambda_{dqs} + p\lambda_{qd0s} \quad (2.1)$$

$$v_{qd0r} = r_r i_{qd0r} + (w_e - w_r)\lambda_{dqr} + p\lambda_{qd0r} \quad (2.2)$$

where

$$(\lambda_{dqs})^T = [\lambda_{ds} \quad -\lambda_{qs} \quad 0]$$

$$(\lambda_{dqr})^T = [\lambda_{dr} \quad -\lambda_{qr} \quad 0]$$

$$(v_{qd0s})^T = [v_{qs} \quad v_{ds} \quad 0]$$

$$(v_{qd0r})^T = [v_{qr} \quad v_{dr} \quad 0]$$

$$(i_{qd0s})^T = [i_{qs} \quad i_{ds} \quad 0]$$

$(i_{qd0r})^T = [i_{qr} \quad i_{dr} \quad 0]$

and relation between flux linkages and currents

$$\begin{bmatrix} \lambda_{qd0s} \\ \\ \lambda_{qd0r} \end{bmatrix} = \begin{bmatrix} K_s L_s (K_s)^{-1} & K_s L_{sr} (K_r)^{-1} \\ \\ K_r (L_{sr})^T (K_s)^{-1} & K_r L_r (K_r)^{-1} \end{bmatrix} \begin{bmatrix} i_{qd0s} \\ \\ i_{qd0r} \end{bmatrix}$$

The matrix $K_s L_s (K_s)^{-1}$ can be expressed as

$$K_s L_s (K_s)^{-1} = \begin{bmatrix} L_{ls}+M & 0 & 0 \\ 0 & L_{ls}+M & 0 \\ 0 & 0 & L_{ls} \end{bmatrix}$$

and

$$K_s (L_{sr})^T (K_s)^{-1} = K_s L_{sr} (K_r)^{-1} = \begin{bmatrix} M & 0 & 0 \\ 0 & M & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

where K_s is given by

$$K_s = \frac{2}{3} \begin{bmatrix} \cos\theta_e & \cos(\theta_e - 120^0) & \cos(\theta_e + 120^0) \\ \sin\theta_e & \sin(\theta_e - 120^0) & \sin(\theta_e + 120^0) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix}$$

After simplifying eq. 2.1 and eq. 2.2 the voltage equations can be written as

$$v_{qs} = r_s i_{qs} + w_e \lambda_{ds} + p\lambda_{qs} \qquad (2.3)$$

$$v_{ds} = ri_{ds} - w_e\lambda_{qs} + p\lambda_{ds} \quad (2.4)$$

$$v_{qr} = r_r i_{qr} + (w_e - w_r)\lambda_{dr} + p\lambda_{qr} \quad (2.5)$$

$$v_{dr} = r_r i_{dr} - (w_e - w_r)\lambda_{qr} + p\lambda_{dr} \quad (2.6)$$

The equations for flux linkages in expanded form

$$\lambda_{qs} = (L_{ls} + M)i_{qs} + Mi_{qr} \quad (2.7)$$

$$\lambda_{ds} = (L_{ls} + M)i_{ds} + Mi_{dr} \quad (2.8)$$

$$\lambda_{qr} = (L_{lr} + M)i_{qr} + Mi_{qs} \quad (2.9)$$

$$\lambda_{dr} = (L_{lr} + M)i_{dr} + Mi_{ds} \quad (2.10)$$

2.2.2 Torque equation in two-phase variables

The torque equation in two-phase variables can be written as

$$T_e = (P/2)[(K_s)^{-1} i_{qd0s}]^T \frac{\partial}{\partial \theta} [L_{sr}](K_s)^{-1} i_{qd0r} \quad (2.11)$$

The above equation in terms of two phase stator flux and stator currents

$$Te = (3/2)(P/2)(\lambda_{ds} i_{qs} - \lambda_{qs} i_{ds}) \quad (2.12)$$

2.2.3 Calculation for slip speed

For controlling the rotor flux of induction motor it is desirable that

$$\lambda_{qr} = 0 \quad (2.13)$$

$$\lambda_r = \sqrt{(\lambda_{dr}^2 + \lambda_{qr}^2)} = \text{Constant} \quad (2.14)$$

Since rotor is squirrel cage

$$v_{dr} = v_{qr} = 0 \quad (2.15)$$

From equations (2.5), (2.6), (2.9), (2.10), (2.15)

$$i_{dr} = 0 \quad (2.16)$$

$$(w_e - w_r). = \text{slip speed} = w_{sl} = (-r_r/\lambda_{dr})i_{qr} \quad (2.17)$$

$$\lambda_{dr} = Mi_{ds} \quad (2.18)$$

$$i_{qr} = -(M/L_r)i_{qs} \quad (2.19)$$

combining equations (2.17,2.18,2.19)

$$w_{sl} = (r_r/L_r)(i_{qs}/i_{ds}) \quad (2.20)$$

where i_{qs} is the torque component of current and i_{ds} is the flux component of current.

2.2.4 Calculation of Three phase Reference Currents

Calculation of three phase reference currents (i_{as}^*, i_{bs}^*, i_{cs}^*) involves the transformation of two phase reference currents from synchronously rotating reference frame to three phase stationary reference frame using a vector rotator (Fig 2.1).

The equation governing the transformation from two phase to three phase

$$i_{abcs}^* = K_s^{-1} i_{qd0s} \tag{2.21}$$

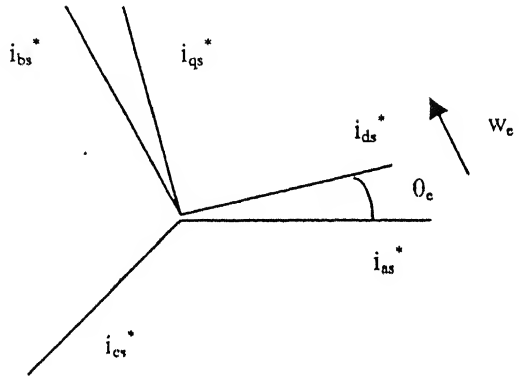


Fig. 2.1

from the Fig. 2.1

$i_{as}^* = i_{qs}^* \cos(90+\theta_e) + i_{ds}^* \cos(\theta_e)$

or,

$$i_{as}^* = i_m \cos(\theta_e + \gamma) \tag{2.22}$$

Similarly,

$$i_{bs}^* = i_m \cos(\theta_e + \gamma + 120^0) \tag{2.23}$$

$$i_{cs}^* = i_m \cos(\theta_e + \gamma - 120^0) \tag{2.24}$$

where

$i_m = \sqrt{[(i_{qs}^*)^2 + (i_{ds}^*)^2]}$

$\gamma = \tan^{-1}(i_{qs}^*/i_{ds}^*)$

The reference currents are fed to the bang-bang controller for controlling inverter switches.

2.3 Control scheme

Functional block diagram of indirect-field oriented control is given in Fig. 2.3. The control scheme basically concentrates on switching the PWM inverter so as to keep rotor flux of induction motor constant. The constant flux operation ensures independent control over torque of induction motor similar to torque control obtained from a separately excited fully-compensated dc motor. As in case of dc motor there are two orthogonal mmfs, one contributes for torque production while other contributes for establishment of working flux. In case of the induction motor this condition is not apparent, so from the point of view of our understanding the stator current vector is divided into two components; one is called flux component of current while other is called torque component of current. Since rotor flux can not be changed with a much faster rate due to large rotor time constant, the flux component of current is kept constant while torque component of current is incremented or decremented based on the torque requirement. The closed-loop control requires automatic adjustment of torque so that drive has good speed response under transient as well as steady-state operation. In the present work the fuzzy controller sets the torque component of current reference depending on magnitude of speed error and change in speed error. The inverter is switched in such a manner that motor currents follow the set current references within a hysteresis band (bang-bang control). The bang-bang logic can be summarized as follows.

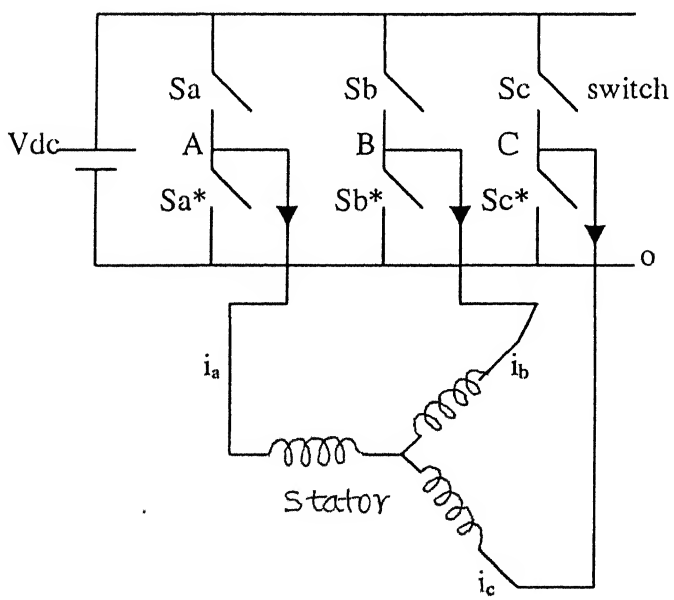
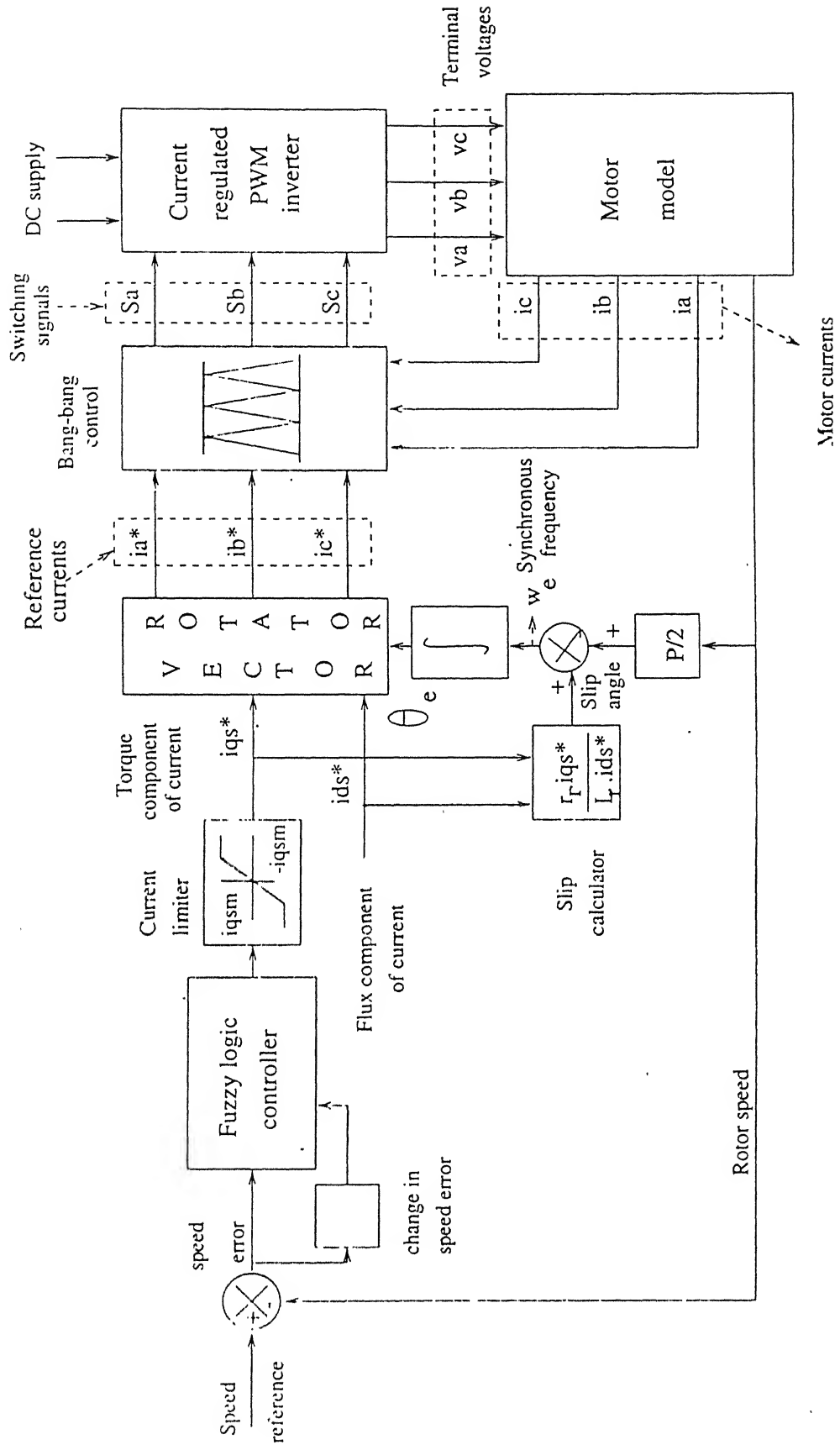


Fig. 2.2 Inverter model



Block diagram of indirect field-oriented control scheme

Fig. 2.3

If i_a, i_b, i_c are motor currents and i_{as}^*, i_{bs}^* and i_{cs}^* are the current reference and h is the hysteresis band

If $i_a < (i_{as}^* - h)$ Sa = 1 and Sa* = 0

If $i_a > (i_{as}^* + h)$ Sa = 0 and Sa* = 1

If $i_b < (i_{bs}^* - h)$ Sb = 1 and Sb* = 0

If $i_b > (i_{bs}^* + h)$ Sb = 0 and Sb* = 1

If $i_c < (i_{cs}^* - h)$ Sc = 1 and Sc* = 0

If $i_c > (i_{cs}^* + h)$ Sc = 0 and Sc* = 1

2.4 Fuzzy logic controller simulation

Fuzzy logic control is utilized for plants with complex dynamics that can not be known precisely. The fuzzy logic control approach for induction motor drive system is very useful since exact mathematical model of the system is not required. The function of fuzzy logic controller is to transform linguistic control rules into control strategy based on expert knowledge.

The fuzzy logic control can be divided into four main functional blocks namely [9]

1. Knowledge base
2. Fuzzification
3. Inference mechanism
4. Defuzzification

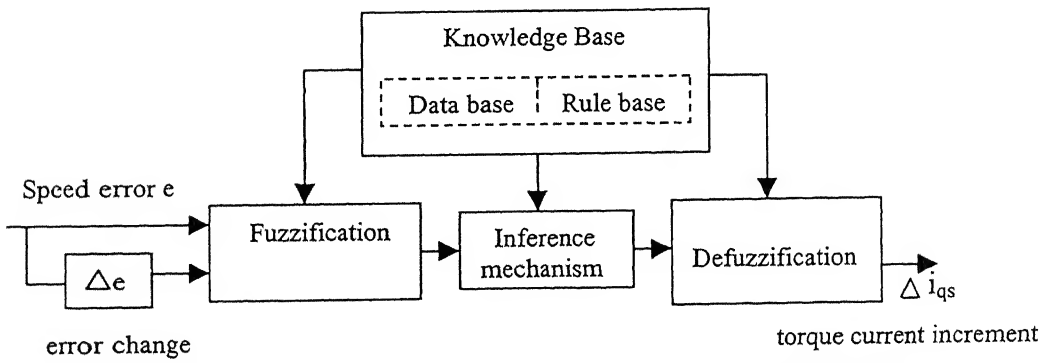


Fig. 2.4 Functional block diagram of fuzzy logic control

The knowledge base is composed of a data-base and a rule base. The data base, consisting of input and output membership functions, provides information for the appropriate fuzzification operations, the inference mechanism and defuzzification. The rule base is made up of a set of linguistic rules relating the fuzzy input variables to the desired control actions. Fuzzification converts a crisp input signal, the error (e), and error change (Δe), into fuzzified signals that can be identified by the levels of membership in the fuzzy sets. The inference mechanism uses the collection of linguistic control rules to convert the input conditions into fuzzified output.

Finally, defuzification converts the fuzzy outputs into crisp control signals, which in the proposed system are the changes in torque component of current (Δi_{qs}^{*}) that drive the induction motor. Design procedure for fuzzy logic controller is as follows.

The fuzzification maps the error, $e(k) = (w_{rs} - w_r)$, and the error change $\Delta e(k) = e(k)-e(k-1)$, to linguistic labels of the fuzzy sets. The proposed system uses following linguistic labels.

For speed error (e)

NL Negative Large

NM Negative Medium

NS Negative Small

Z Zero

PS Positive Small

PM Positive Medium

PL Positive Large

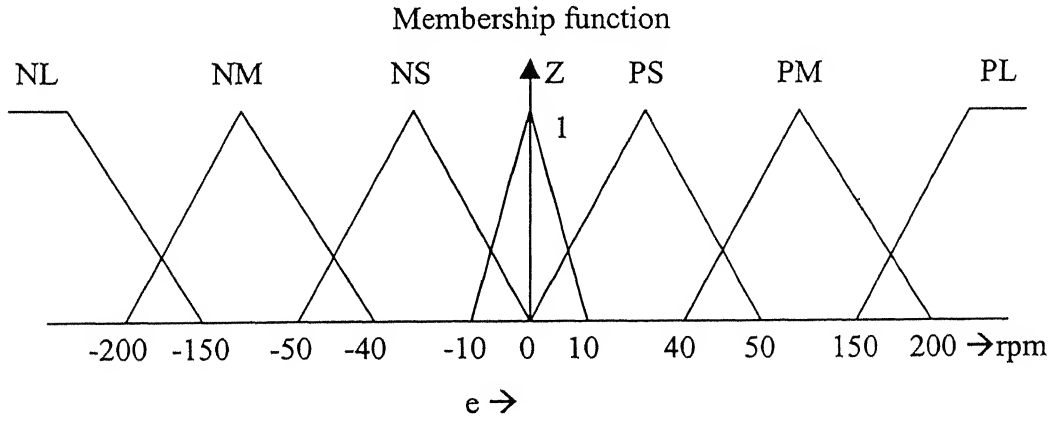


Fig. 2.5 Membership function for speed error

For speed error change Δe

NL Negative Large

NS Negative Small

Z Zero

PS Positive Small

PL Positive Large

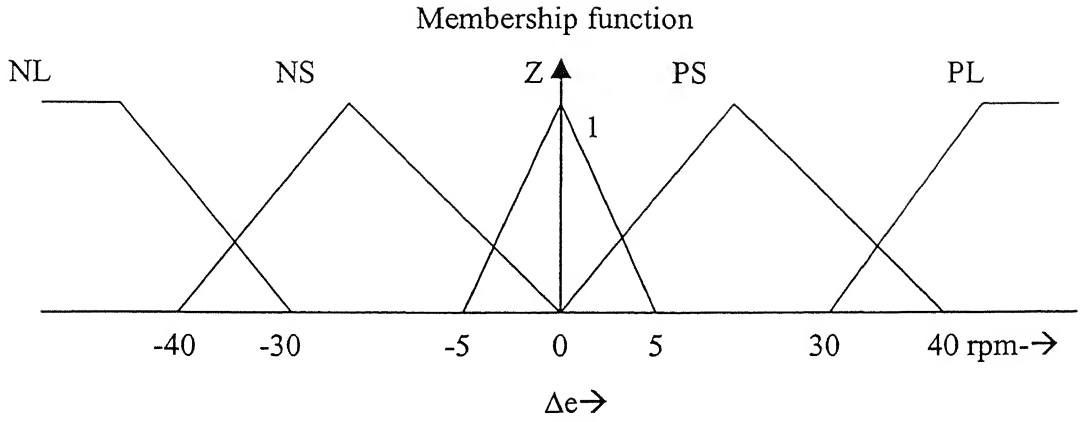


Fig. 2.6 Membership function for change in speed error

For output current change Δ i_{qs}

NL Negative Large

NS Negative Small

Z Zero

PS Positive Small

PL Positive Large

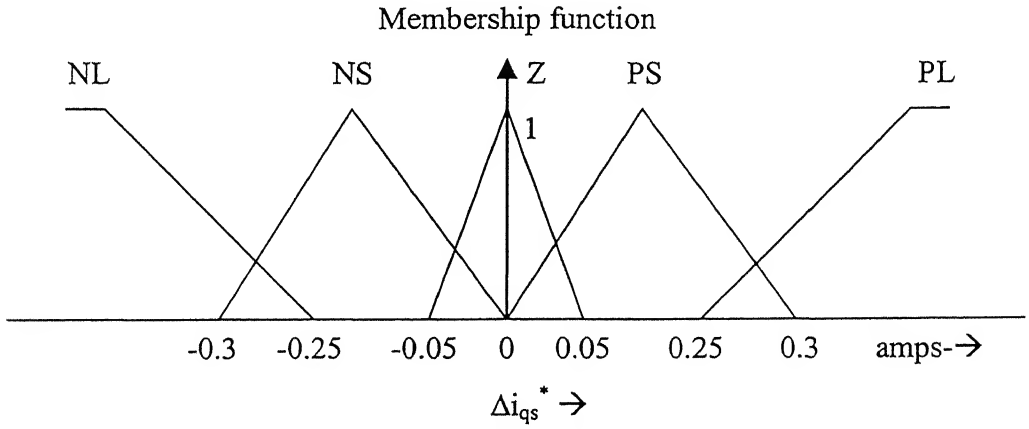


Fig. 2.7 Membership function for change in torque component of current

2.5 Control rules

Knowledge base involves defining the rules represented as statements governing the relationship between input and output variables in terms of membership functions. The control rules are represented as a set of if then rules, for example if speed error is PL and error change is Negative Large then change in output is Positive Large. Table2.1 shows the various fuzzy control rules.

The control rules are evaluated by an inference mechanism. During the rule evaluation step, the combination of selected rules in knowledge base is evaluated. There are several mechanism for implementation of inference mechanism, such as min-max algorithm, the correlation algorithm etc [9]. The present work uses **Mamdani algorithm** for drawing inference.

The output generated by fuzzy controller, which is used to control the plant, must be crisp value according to fuzzy output from inference mechanism. The defuzzification block does this job i.e. it transforms crisp output in accordance with fuzzy output generated by inference mechanism. Many defuzzification strategies are available [9], such as, the weight average criterion, the mean-max (mean of maximum) membership, and center-of–area method (Centroid method). The **center of area** method is used here for defuzification, which can be summarized as follows.

Suppose speed error is –5 and change in speed error is –10, then e will be NS or Z and change in error will be NS. So the rules relating the error and error rate are

If e is NS and Δe is NS then change in output current is NS (Fig. 2.7).

If e is Z and Δe is NS then change in output current is PS (Fig. 2.8).

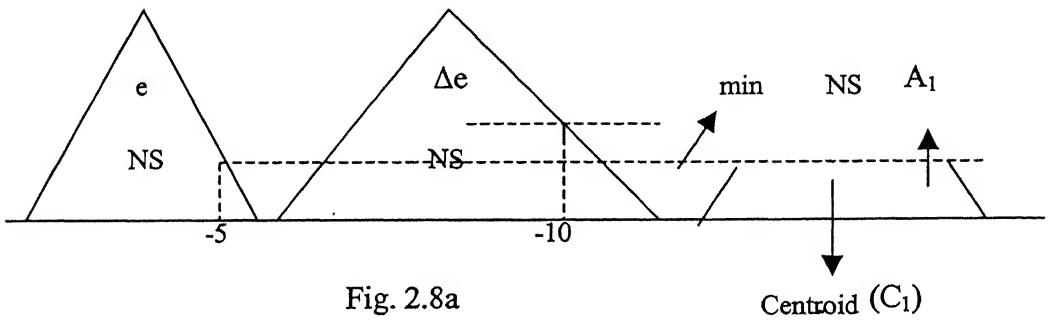


Fig. 2.8a

Table 2.1 Control Rules

If speed error is e	and error change is Δ e	Then increment in output current is Δi_{qs}^*
PL		PL
PM	NL	PL
PM	NS	PL
PM	Z	PS
PM	PS	PS
PM	PL	NS
PS	NL	PL
PS	NS	PS
PS	Z	PS
PS	PS	PS
PS	PL	NS
Z	NL	PS
Z	NS	PS
Z	Z	Z
Z	PS	NS
Z	PL	NS
NS	NL	PS
NS	NS	NS
NS	Z	NS
NS	PS	NS
NS	PL	NL
NM	NL	PS
NM	NS	NS
NM	Z	NS
NM	PS	NL
NM	PL	NL
NL		NL

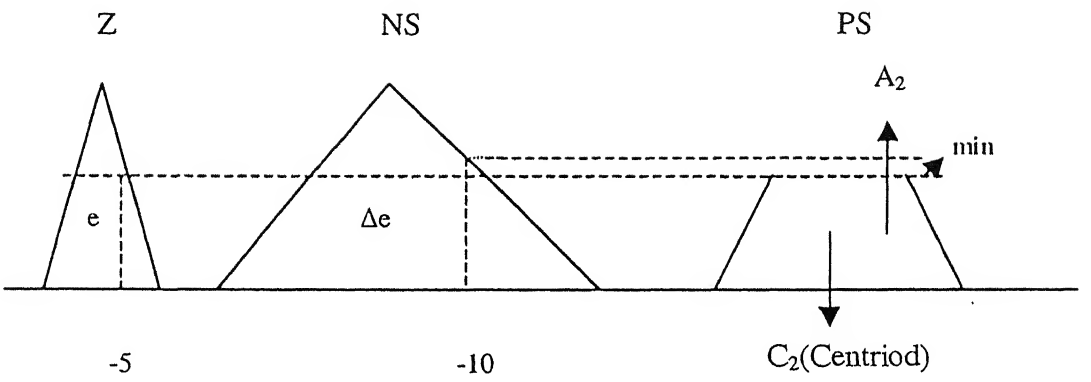


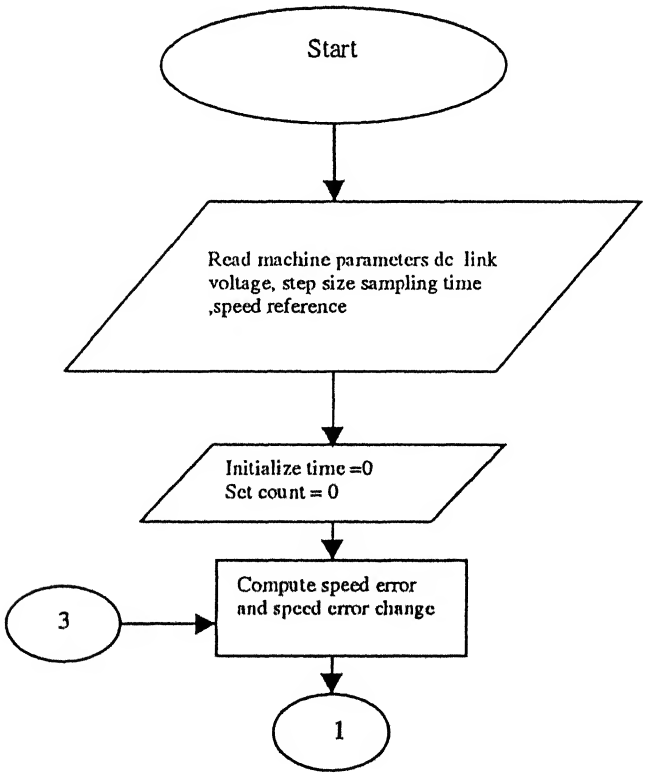
Fig.2.8b

Center of area gives the defuzzified value of current increment as

$$C12 = (C_1A_1 + C_2A_2)/(A_1 + A_2)$$

2.6 Computer simulation of the drive system

This section gives flowchart of the computer program developed for simulating the performance of the induction motor drive.



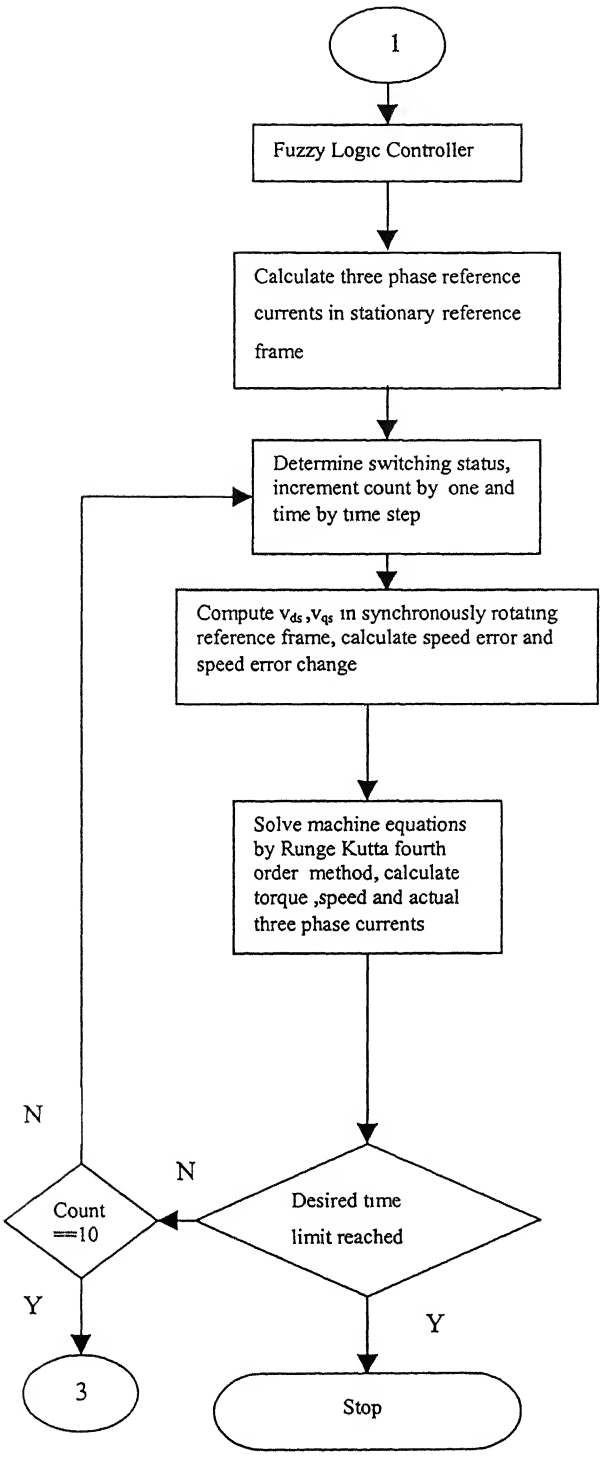


Fig. 2.9 Flow chart for the simulation program

The proposed system has been simulated in 266 MHz pentium pc in Linux environment. The simulation program is written in C and the flowchart is given in Fig. 2.9. The complete C-program listing is included in Appendix B.

2.7 Simulation results

The drive has been given a step change in speed command and a step change in torque command to demonstrate feasibility of proposed scheme under varying operating conditions of operation. Computer traces of rotor flux response under change in speed command and torque command are shown in Fig. 2.10. The effectiveness of fuzzy controller is validated by comparing the results with that obtained using a conventional PI controller.

2.7.1 Rotor flux response under change in speed command

Fig. 2.10a, 2.10c show rotor flux response and rotor speed response when a step change of −1000 rpm (speed reversal)is applied at t =1.0s. Initially induction motor is allowed to run at zero speed by setting speed command to zero. At t = 0.1s speed command is set at +500 rpm and further at t = 1.0s speed command is set at −500rpm. From Fig. 2.10a, rotor flux is settling at its steady state value in 0.2s from its value at standstill again in the region of speed reversal a slight overshoot is seen in rotor flux, in this region machine is operating in regenerative braking mode. Rotor speed response (shown in Fig. 2.10c) shows that machine settles at +500 rpm within 0.1s and from +500 rpm to −500rpm in 0.2s. Fig. 2.10b shows locus of rotor flux in d-q plane under change in speed command, the circular nature of rotor flux vector in d-q plane ensures the vector control operation of induction motor drive system in transient as well as in steady state.

2.7.2 Rotor flux response under change in torque command

Fig. 2.10d, 2.10e, 2.10f show rotor flux response under change in torque command applied to the proposed system. Initially machine is allowed to run at +500 rpm by setting speed command to +500 rpm and torque command to zero. At t =1.0 s. a step of +5.0 N.m is applied. From Fig. 2.10d, it can be seen that magnitude of the rotor flux is constant in spite of applied torque. The circular nature of rotor flux vector (shown in Fig. 2.10e) in d-q plane ensures vector control operation of induction motor drive system. Torque response (shown in

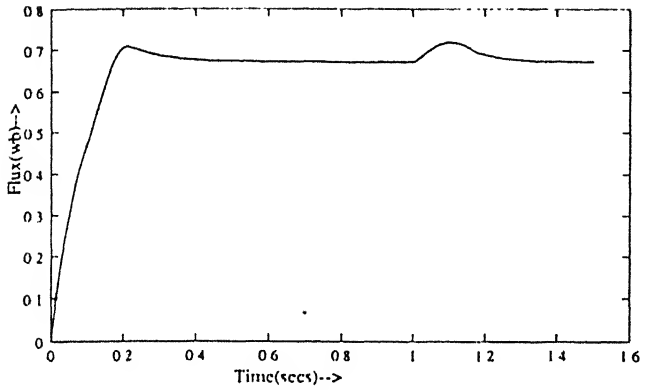
Fig. 2.10f) shows that electrical torque of machine settles at set command within a few supply-cycle thereby showing excellent torque response of the proposed system.

2.7.3 Comparison

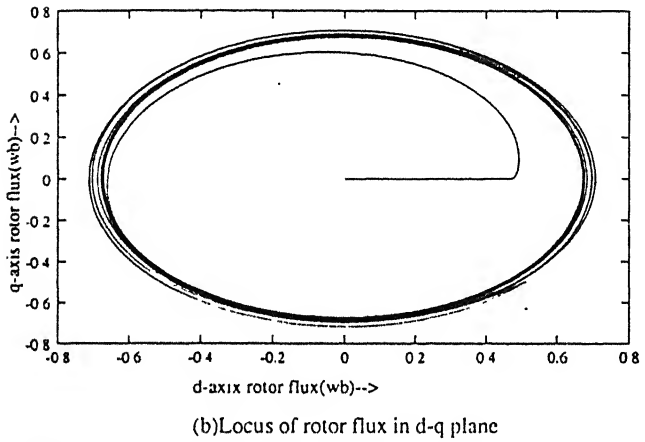
Torque response and speed response of the proposed system is compared with the results obtained from operation of proposed system with PI controller. Simulated results are taken under change in speed command from +500 rpm to –500 rpm. Fig. 2.11 shows flux, torque and speed response when induction motor drive is controlled by a fuzzy controller, and Fig. 2.12 shows flux, torque and speed response when induction motor drive is controlled by a PI controller. As can be seen from rotor speed response (Fig. 2.11c and Fig. 2.12c) that under drive operation with fuzzy controller rotor settles from +500 rpm to –500 rpm within 0.2 s whereas this figure is approximately 1.0 s for PI controller. Further from torque response curve (Fig. 2.11b and Fig. 2.12b) it can be concluded that torque response of the fuzzy controlled drive system is very fast as compared to that of the conventional PI controller. This discussion demonstrates the effectiveness of fuzzy controller for controlling non-linear systems like induction motor.

2.8 Conclusion

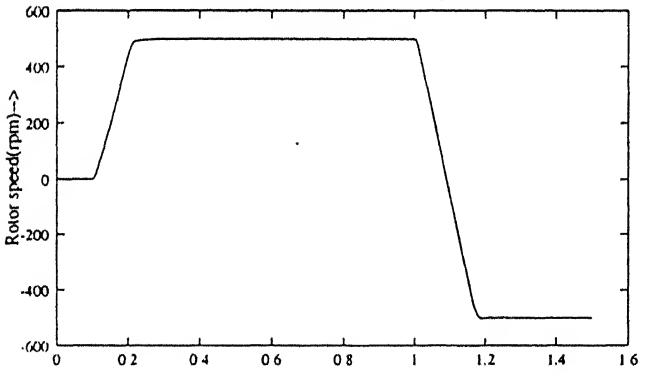
The proposed indirect field oriented control scheme is studied by computer simulation. The simulated results show that fuzzy logic control can give fast torque response under transient conditions. The comparison of simulated results with that obtained from conventional PI controller shows that fuzzy logic control is very fast.



(a)Rotor flux

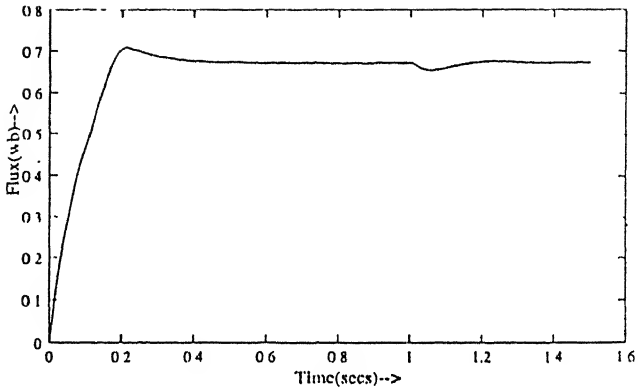


(b)Locus of rotor flux in d-q plane

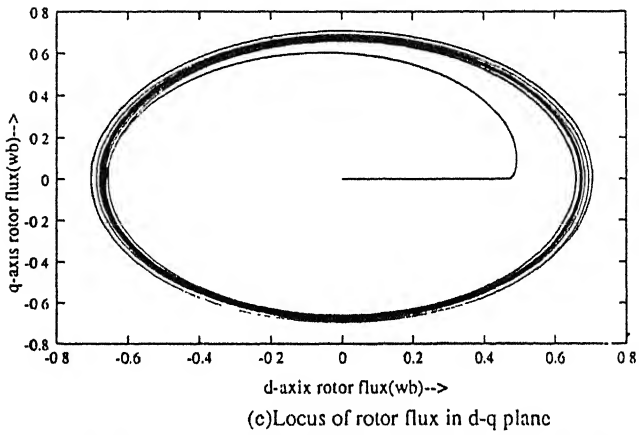


(c) Rotor speed

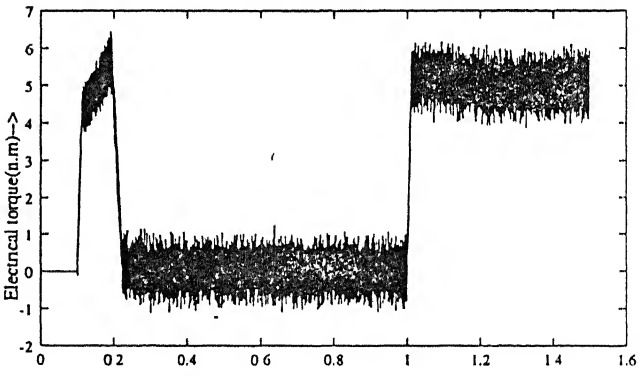
Waveform a,b,c with step change of -1000 rpm at t=1.0s



(d)Rotor flux



(c)Locus of rotor flux in d-q plane

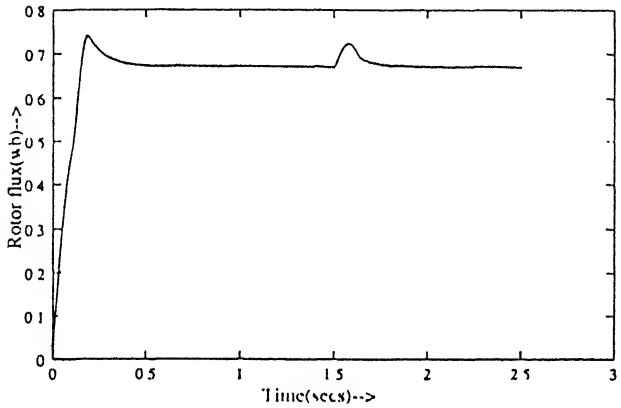


(f) Electromagnetic torque

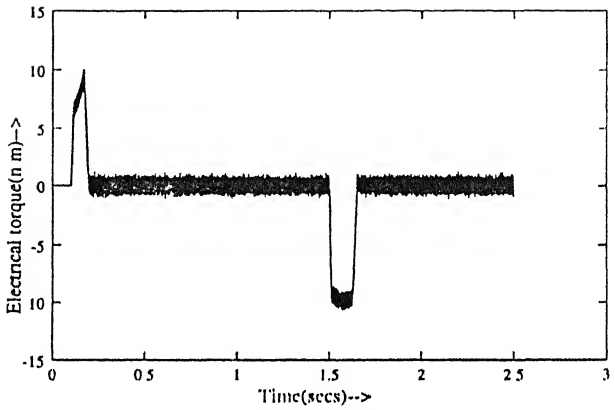
Waveform d,e,f with step change of -1000 rpm at t=1.0s

Rotor flux waveform under change in speed and torque command

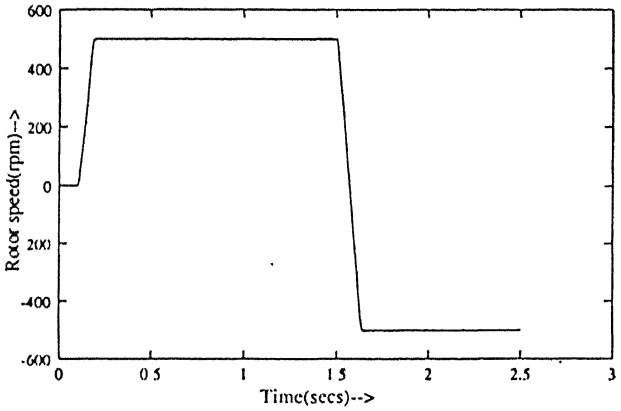
Fig. 2.10



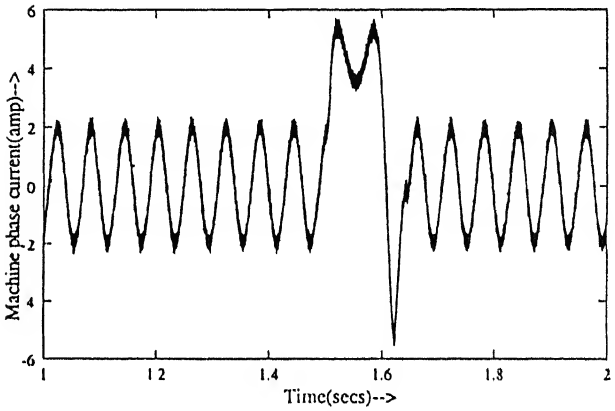
(a) Rotor flux



(b) Electromagnetic torque

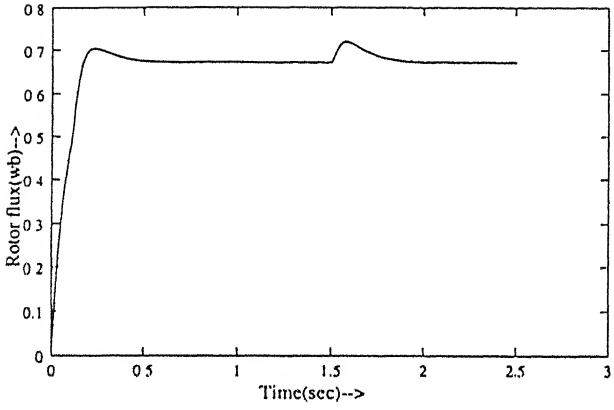


(c) Rotor speed

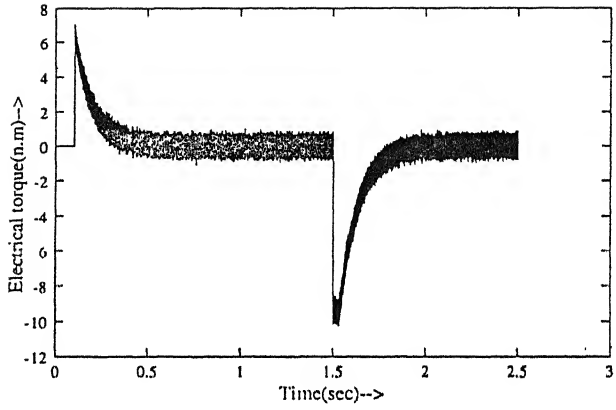


(d) Machine phase current

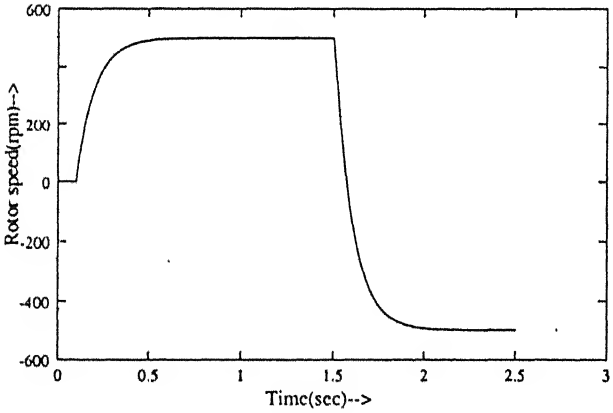
Fig. 2.11 Simulated results obtained from proposed system with controller as fuzzy
(Step change of -1000 rpm at t = 1.5s)



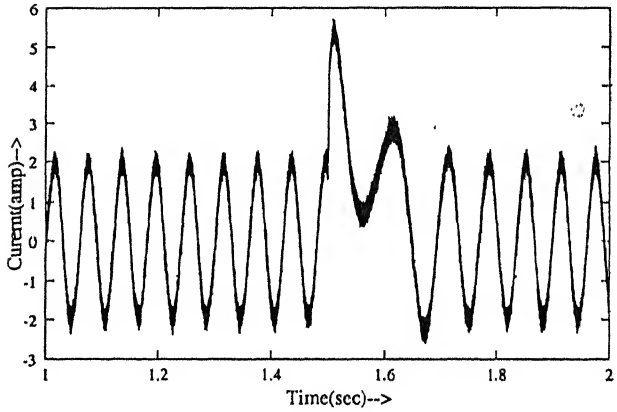
(a)Rotor flux



(b)Electrical torque



(c)Rotor speed



(d)Machine phase current

Fig. 2.12 Simulated results obtained from the proposed system with PI controller
(Step change of -1000 rpm at t=1.5s)

Chapter 3

Hardware and Control Software Implementation

3.1 Introduction

The proposed system has been practically implemented with the help of a personal computer having a high performance data acquisition card (PCL-208). The data acquisition card provides a channel for communication between software and hardware. There are some distinct advantages offered by PC-based implementation such as user friendliness, ease of programming and availability of large memory for storage. Additionally PCs have standard architecture so a software written for a particular PC can be run on other PC manufactured by some other firm. This type of flexibility is not present with micro-controller systems, where a software written for a particular micro-controller system may not be executed on other micro-controller systems.

The block diagram of the proposed system is shown in Fig. 3.1. Various blocks such as error generator, fuzzy controller, torque current limiter, vector rotator, slip frequency calculator and pulse width modulation logic has been implemented in real time by writing a control program in C which is given in the Appendix C.

3.2 Specification of data acquisition card

The specification of data acquisition card is as follows [12].

1. Switch selectable 16 single-ended or 8 differential analog input channels.
2. An industrial standard 12 Bit successive approximation converter (ADC674) to convert analog inputs. The maximum A/D sampling rate is 60 KHz in DMA mode.
3. Switch selectable versatile analog input ranges.

Bipolar: +/- 0.5 V, +/- 1 V, +/- 2.5 V, +/- 5 V, +/- 10 V.

Unipolar: + 1 V, + 2 V, + 5 V, + 10 V.

4. Provide three A/D trigger modes: software trigger, programmable pacer trigger and external trigger pulse trigger.
5. A/D converted data can be transferred by program control. interrupt handler routine or DMA transfer.
6. An INTEL 8254 Programmable Timer/ Counter provides pacer output (trigger pulse) at the rate of 2.5 MHz to 71 minutes/ pulse to the A/D. The timer time base is switch selectable 10 MHz or 1 MHz. One 16-bit counter channel is reserved for user configuration applications.
7. Two 12 bit monolithic multiplying D/A output channels. Output range of 0 to +5 V can be created by using the onboard –5 V reference. This precision reference is derived from the A/D converter reference. External AC or DC reference can also be used to generate other D/A output ranges.
8. TTL/ DTL compatible 16 digital input and 16 digital output channels.

3.3 Setting of the data acquisition card

The setting of card for present application is as follows:

1. Base address is set at 0x300 and corresponding I/O range is from 300 to 30F hexa-decimal.
2. Analog input and analog output are set in bipolar mode and range of output voltage is ± 5V.
3. Analog input channels are set in single ended mode.
4. The clock frequency of 8254 is set at frequency of 10 MHz.

For inputting actual speed, three phase actual currents and speed reference to the software five single ended analog input channels have been used. The actual currents are compared with the reference currents and the PWM logic is generated. The PWM signals to the upper transistors of the inverter are taken out by means of three digital output ports, which give +5V output corresponding to ON position of switch. The firing pulse to lower switch of a leg is generated by complementing pulse of upper switch. In real time the complement of pulse of upper switch is obtained by using NOT gates. The use of NOT gates also ensures protection against any programming mistake, which may simultaneously ON upper and lower switch of a leg.

3.4 Lockout circuit

Power devices require a finite amount of time to turn-off and turn-on. At the time of turn-on of upper switch and turn-off of lower switch of a leg, there exist an interval in which both devices conduct. This type of situation should not exist in real time application since simultaneous conduction of both switches of a leg will short circuit the DC link voltage giving rise to high current thereby damaging power devices. To avoid this situation a sufficient amount of blanking time should be provided between turn-on of upper switch and turn-off of lower switch. In real time this blanking time has been provided by using 74123 mono-stable multivibrator, which generates proper time delay governed by externally connected resistance and capacitance. In the proposed scheme a lock out time of 20 μ s has been chosen. The circuit diagram of lock out circuit is shown in Fig. 3.2.

3.5 IGBT gate drive circuit

Power devices require large amount of gate currents for turning them ON and OFF, this requirement can not be met directly by outputs generated by low power control circuit, which provides pulses of 0-5V. The large gate current requirement is met by amplifying outputs of low power control circuitry. A pulse amplifier, which generates $\mp$ 12V output corresponding to 0-5V generated by control circuit, has been used for experiment. Pulse amplifier can be designed to detect any over current present in the device by monitoring collector voltage. Firing circuit (shown in Fig. 3.3) used here gives over current indication by lighting up an LED.

3.6 Motor current sensing

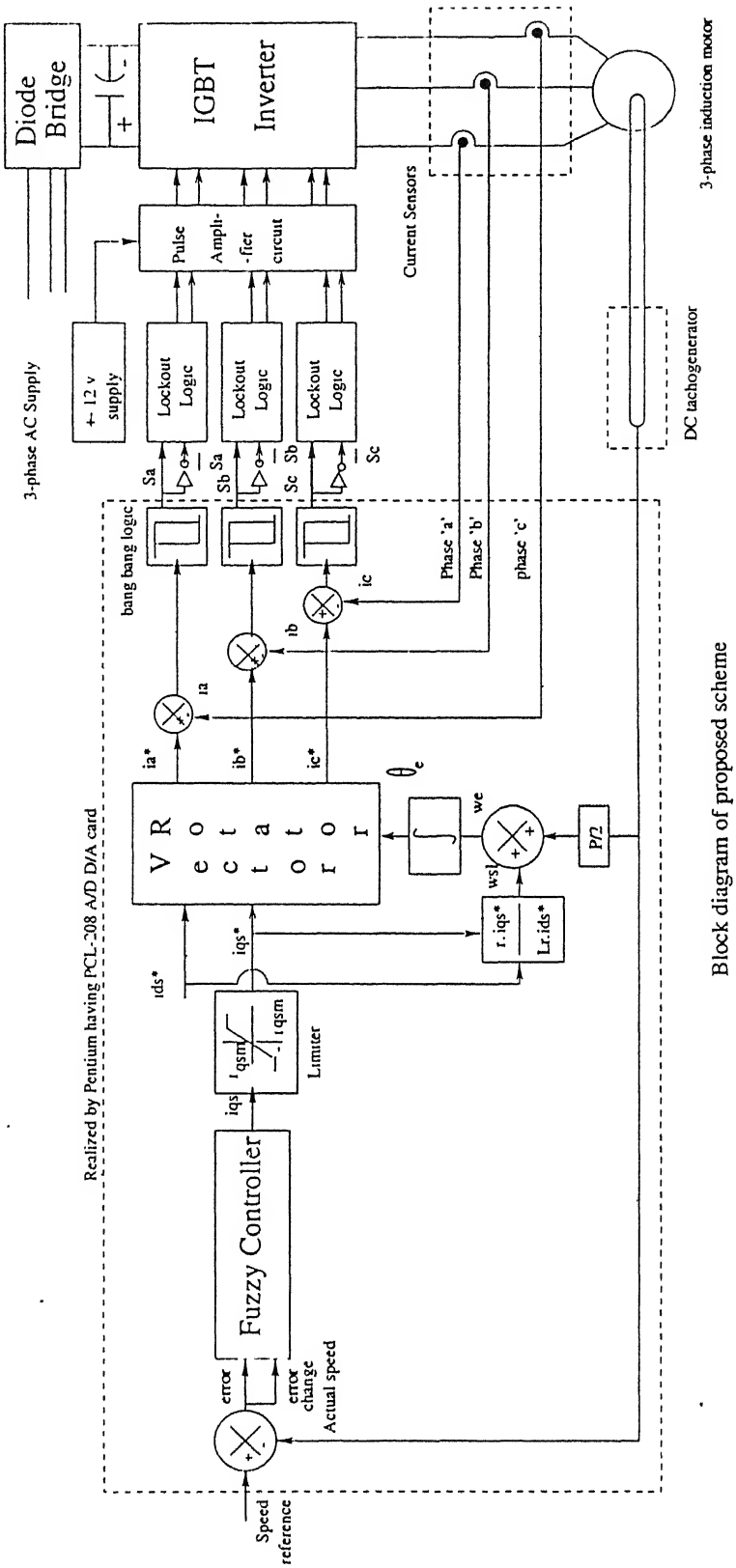
Actual current is sensed by Hall effect current sensor, which gives output at its measuring terminals in proportion with actual current. Use of Hall effect current sensor for current sensing also ensures isolation between high power circuit and low power control circuit. Output of current sensor used in experimental work, is .63V/amp. The detailed specification of the current sensor is given in Appendix A.

3.7 Control software

The control software gives command to the processor to read input data from specified hardware port and send certain output data to the specified hardware port. In the present application control software reads five input data from five analog input channels. These input data are used for determining the status of switching signals and finally these firing signals are outputted to specified hardware port. The execution time for reading five inputs from hardware port and sending three outputs to hardware port is 90 micro seconds and clock speed of pentium processor is 266 MHz. The flowchart for deriving control program is given in Fig. 3.4. The control program is listed in Appendix C.

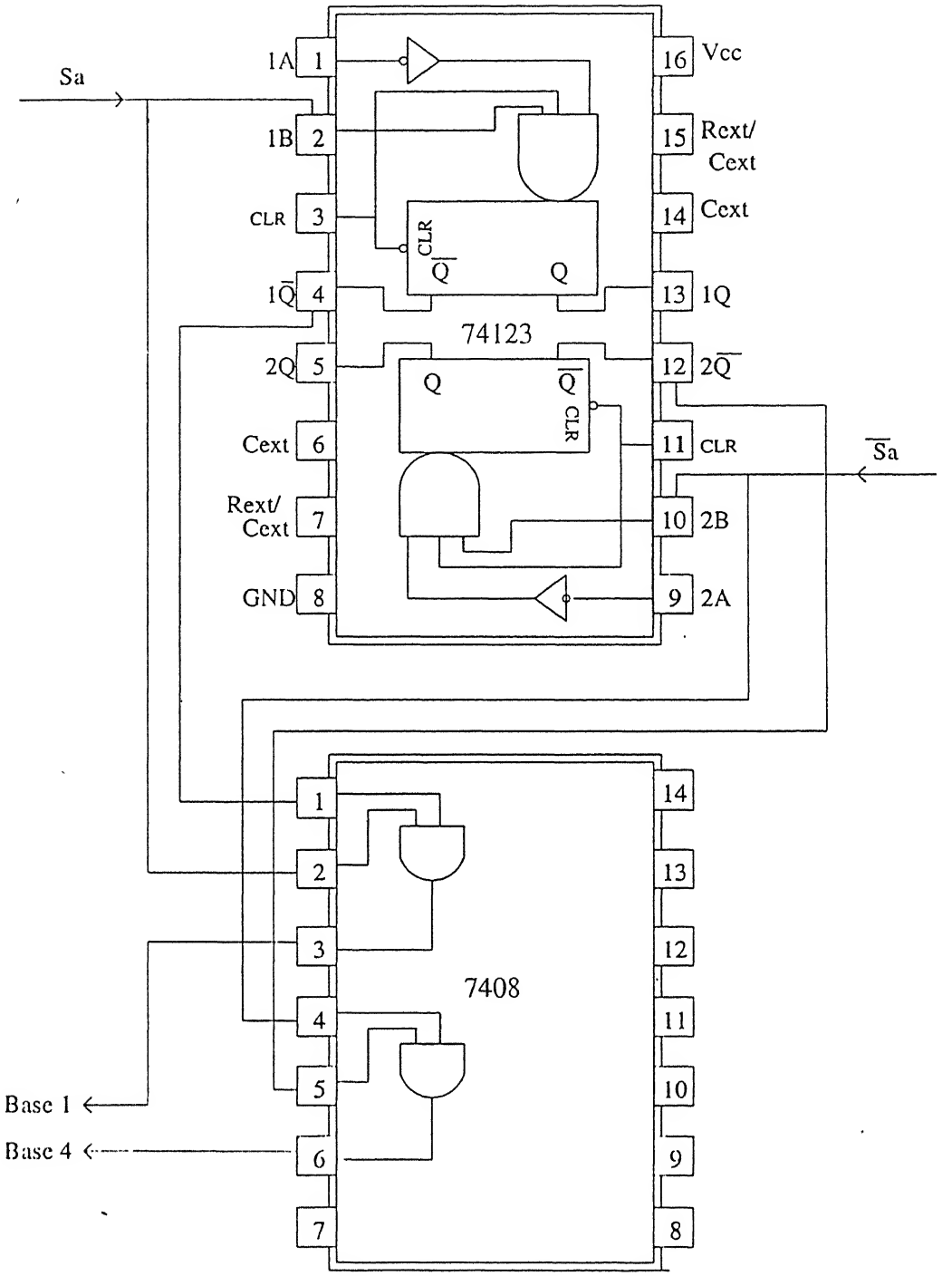
3.8 Conclusion

The proposed system is implemented on a three phase induction motor (specifications are given in Appendix A). The experimental results obtained confirm the feasibility of the proposed system.



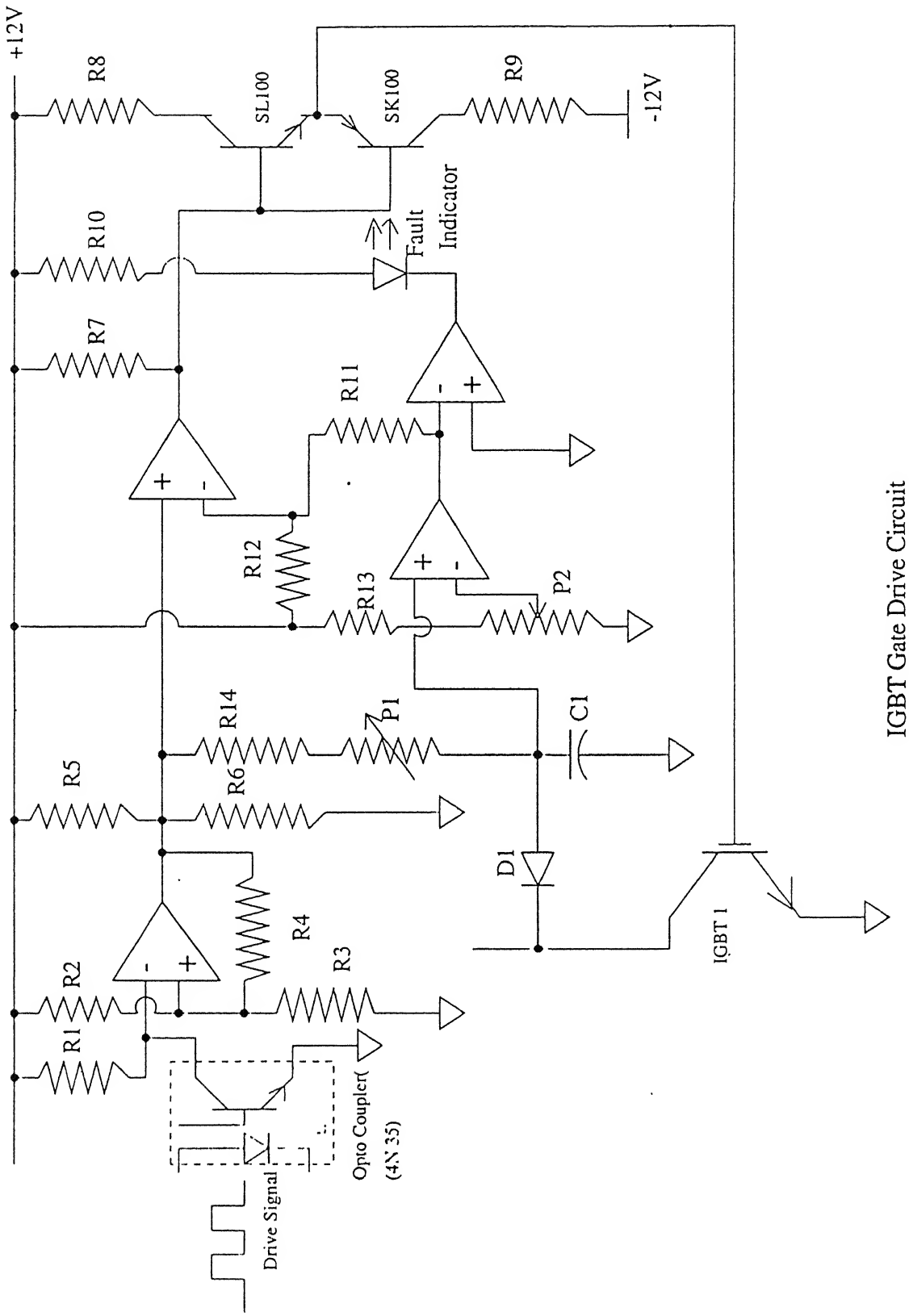
Block diagram of proposed scheme

Fig. 3.1



A view of lock out circuit for the leg of phase a

Fig. 3.2



IGBT Gate Drive Circuit

Fig. 3.3

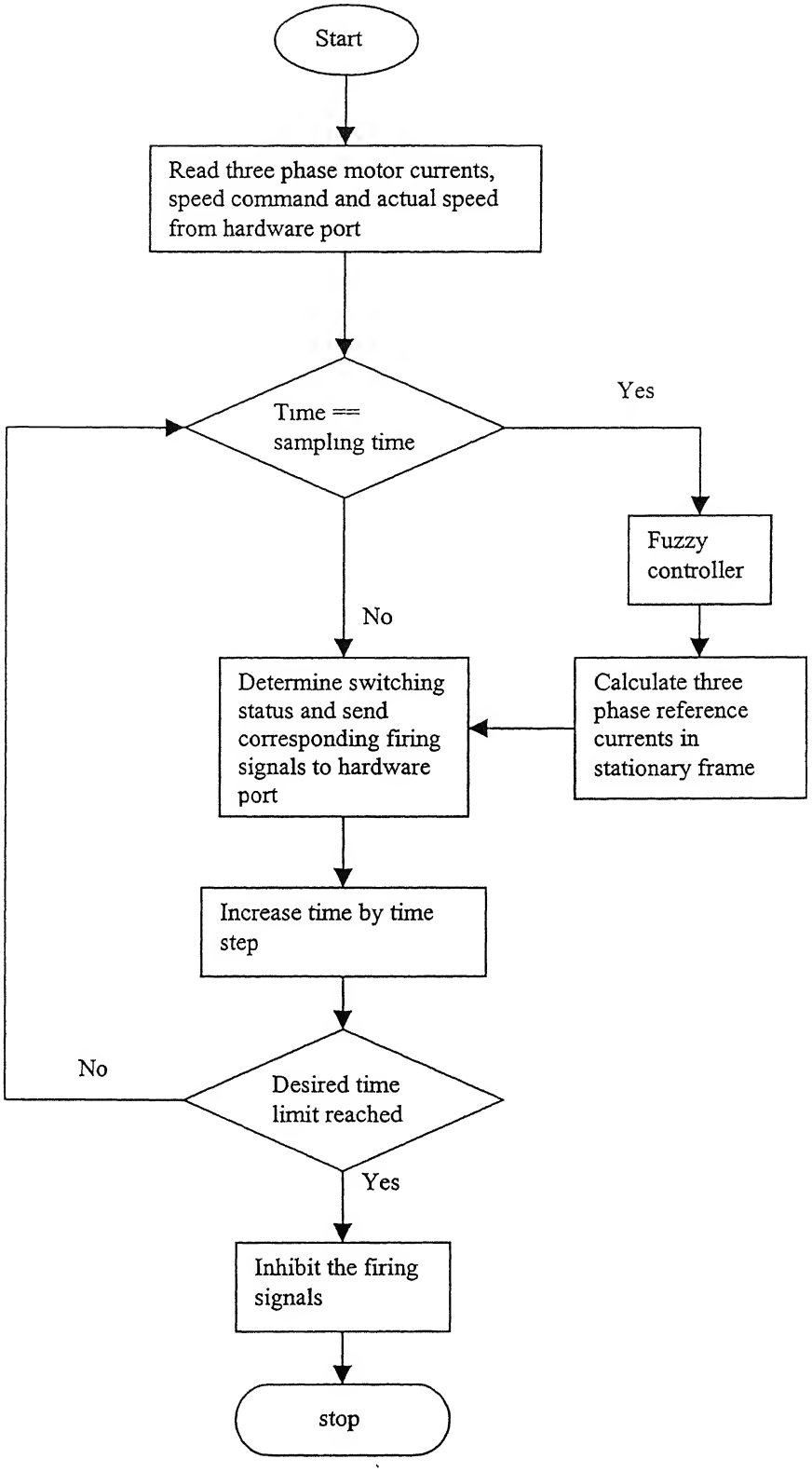


Fig. 3.4 Flow-chart for the control program

Chapter 4

Experimental and Simulation Results

4.1 Introduction

The proposed scheme has been tested on a 3-phase, 415 V star connected induction motor, which is mechanically coupled with a 230 V, 1500 rpm dc generator for loading the induction motor indirectly. The magnitude for various parameters for inverter, motor and the dc generator are given in Appendix A.

4.2 Speed response

The proposed system is given a step change in speed from +500 rpm to –500 rpm. Simulated speed response, rotor flux response and torque response during speed reversal is shown in Fig. 4.1. For t < 0.1 s machine is allowed to run at zero speed by setting speed reference to zero, at t = 0.1 s reference speed is set at +500 rpm, further at t = 1.0 s reference speed is set at –500 rpm. From simulated speed response it can be seen that machine settles at +500 rpm from standstill in approximately 0.1 s, and at –500 rpm from +500 rpm within 0.2 s. From t = 1.0 s to t = 1.2 s machine is operating in regenerative braking mode. From rotor flux response it can be seen that rotor flux settles in steady state value within 0.2 s, during speed reversal rotor flux has slight overshoot and after reversal rotor flux again settles at its steady state value. From the torque response curve, it can be seen that drive has very fast torque rise or fall during transient conditions, pulsations present in torque is inherent in pulse width modulated inverter fed drives. Speed response and machine current obtained from experiment are shown in Fig. 4.2 and Fig. 4.3, From which it can be seen that they are in close agreement with the simulated speed and current responses.

4.3 Torque response

The induction motor in proposed system is subjected to a step change in torque command of +5 N.m at t = 1.0 s, from the simulated torque response (shown in Fig. 4.4) it can be seen that

torque response is very fast. Rotor flux response shows that magnitude of rotor flux is constant under transient as well as in steady state conditions. Speed response shows that sudden application of torque causes a slight dip in speed but within a few time motor again settles at set speed. In the experiment the load test is performed on the induction motor by driving a dc generator feeding a resistive load, mechanically coupled to the induction motor. The applied torque to the machine is measured by measuring armature current of dc generator. In the proposed scheme applied torque is 3.37 N.m. Experimental speed response shows that machine follows the set speed response after application of sudden load. Simulated speed response and practical speed response (shown in Fig. 4.5) are found to be in close agreement.

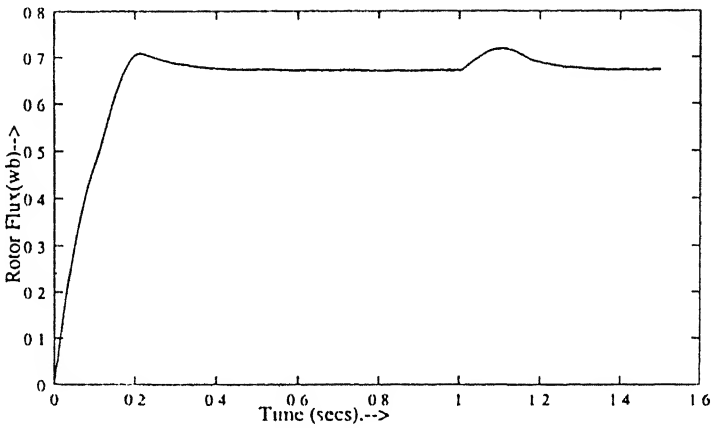
4.4 Current response

Simulated current response and practical current response are shown in Fig. 4.6 and Fig. 4.7. From the simulated and practical current responses it can be seen that actual currents are following the set current reference within a set hysterisis band. The phase voltage waveform of the induction motor is shown in Fig. 4.8.

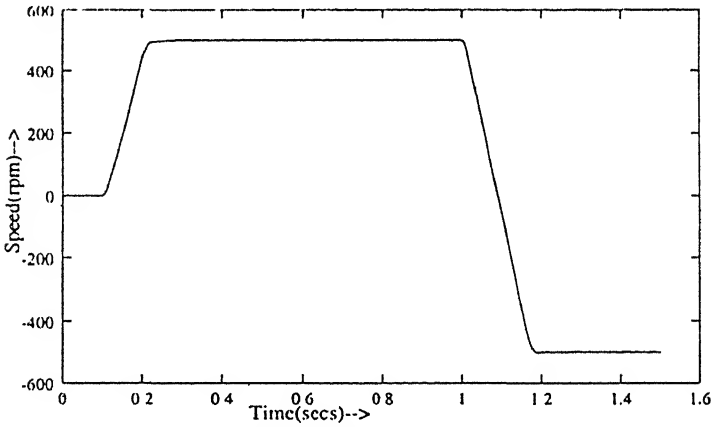
4.5 Conclusion

From the simulated and practical results obtained, following conclusions can be drawn.

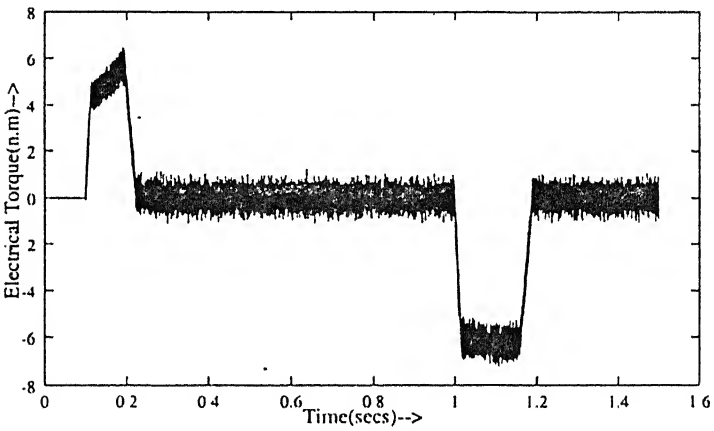
1. The induction motor is operating under constant rotor flux in steady state as well as in transient conditions, such as sudden application of step torque or application of step speed command.
2. The constancy of rotor flux during transient and steady state indicates that machine remains under vector control in transient as well as in steady state conditions.



(a) Rotor flux



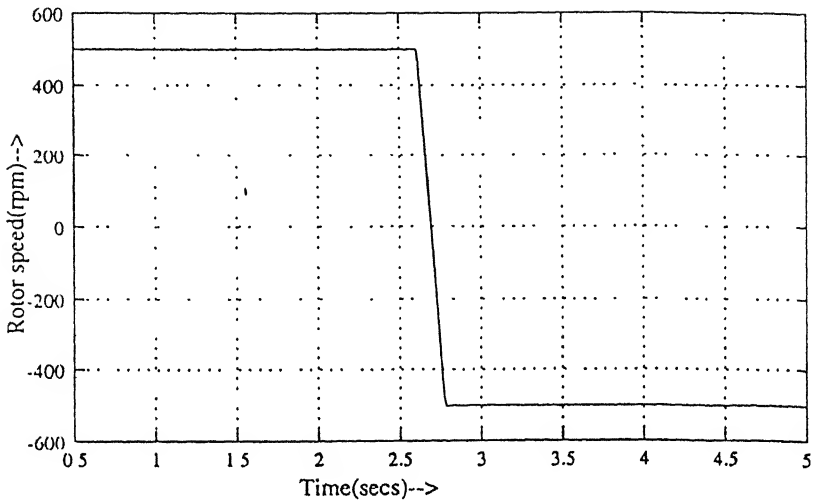
(b)Rotor speed



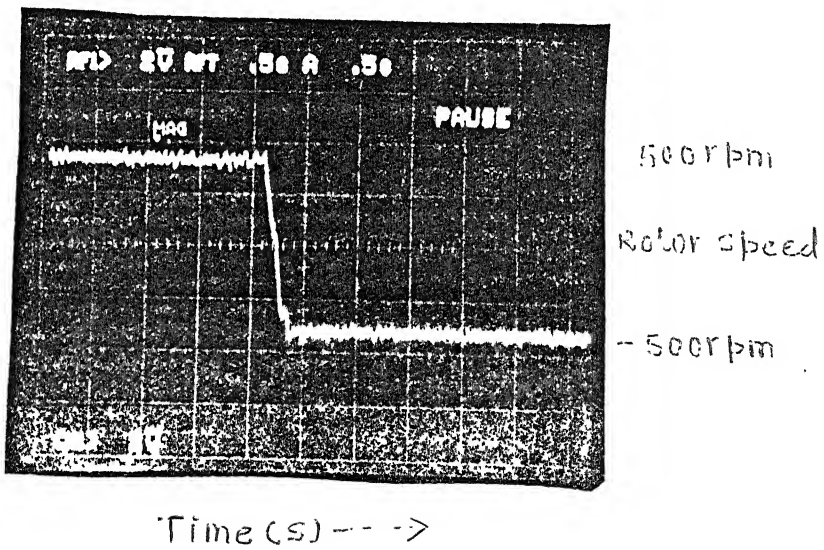
(c)Electromagnetic Torque

waveforms with a step change of -1000 rpm at t=1.0s

Fig. 4.1



(a)Rotor speed during speed reversal from +500rpm to -500 rpm



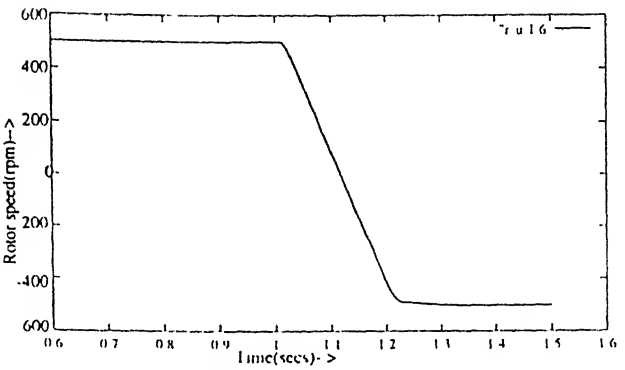
(b) Rotor speed during speed reversal from +500rpm to -500 rpm

Sensitivity of speed sensor: .0406v/rpm

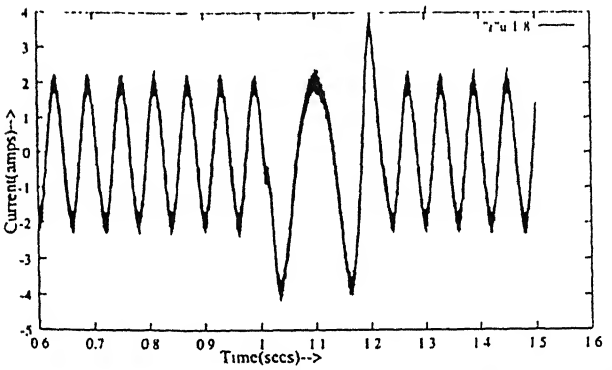
Rotor speed waveforms : (a) Simulated

(b) Experimental

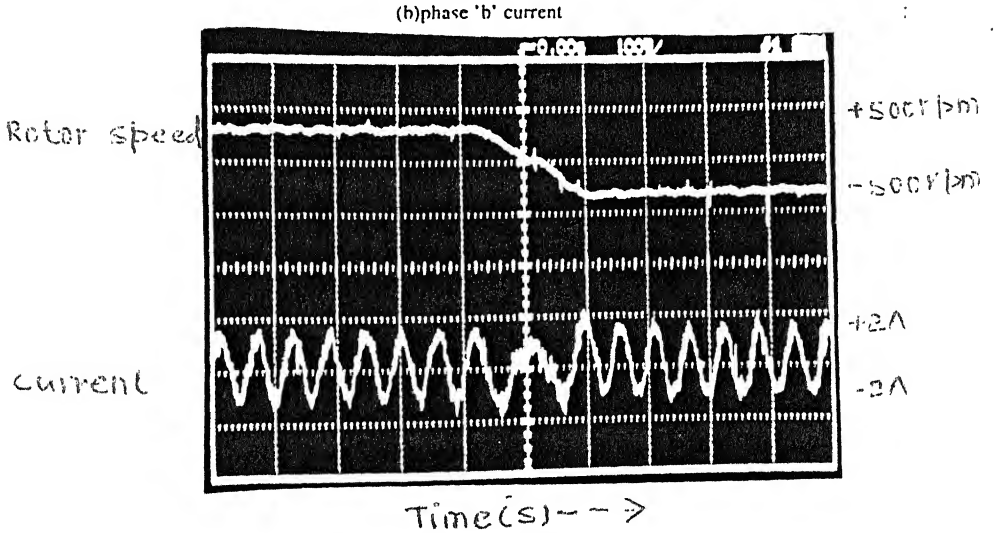
Fig. 4.2



(a)Rotor speed



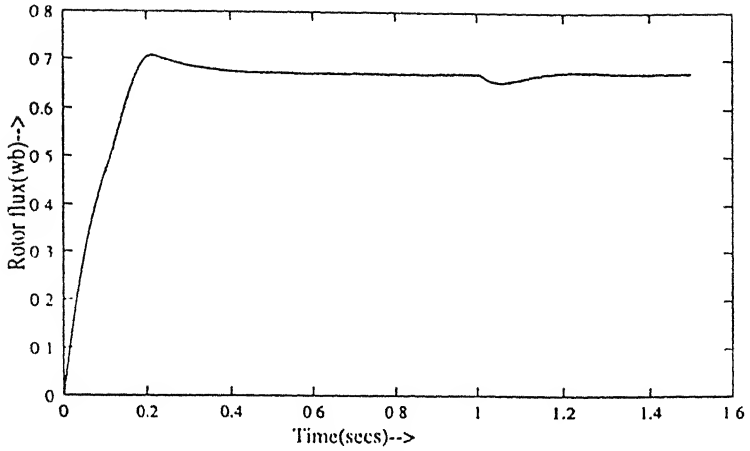
(b)phase 'b' current



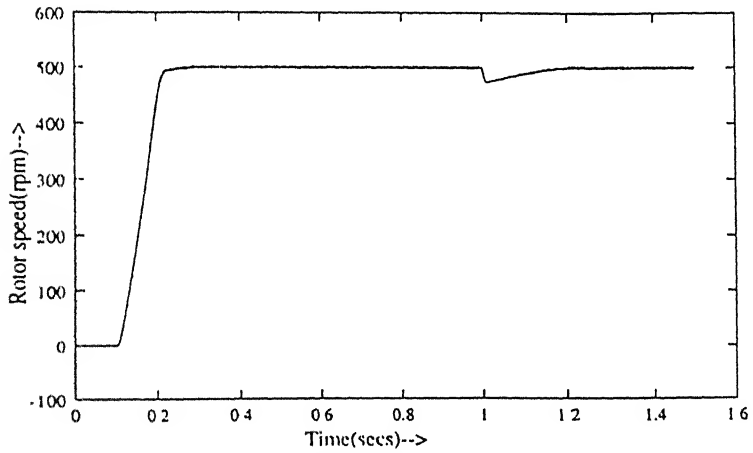
(c) rotor speed and phase 'a' current waveforms
scale :5.0v/div for upper waveform
scale: 2.0v/div for lower waveform

Speed and current waveforms : a,b- Simulated
c-Experimental

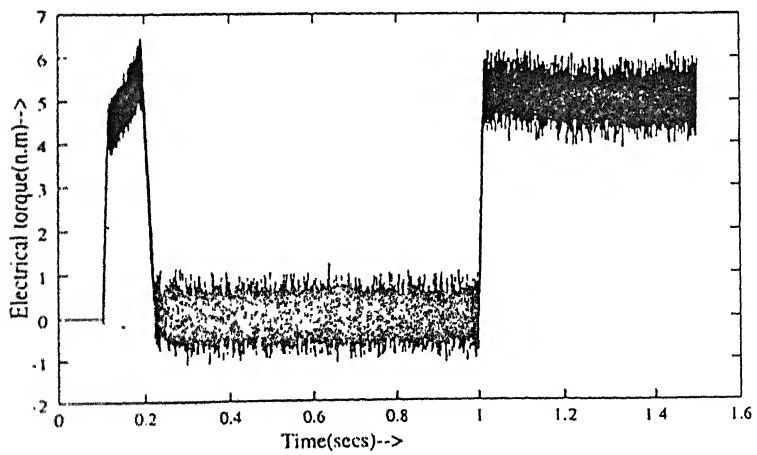
Fig. 4.3



(a)Rotor flux



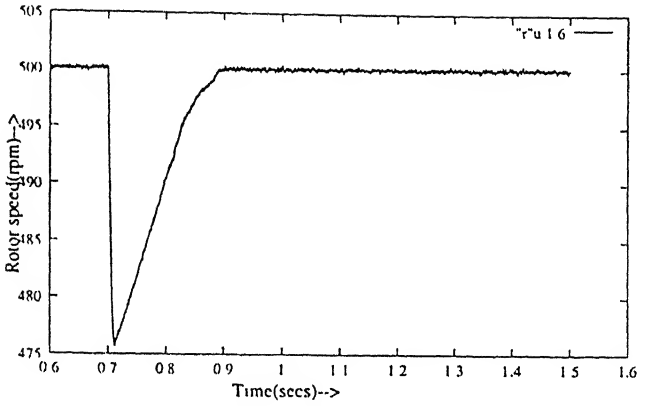
(b)Rotor speed



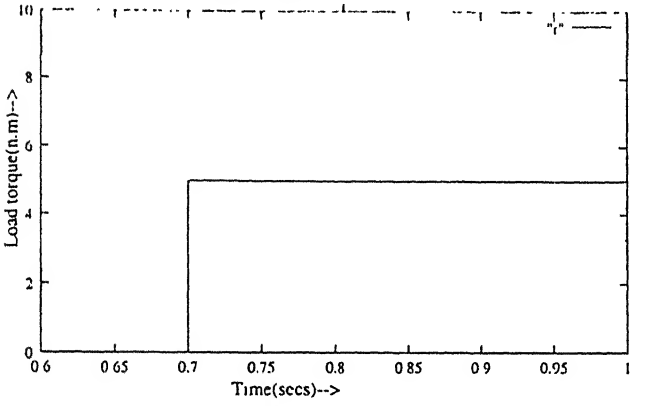
(c)Electromagnetic torque

Waveforms a,b,c with a step change of +5.0 N.m at t=1.0sec.

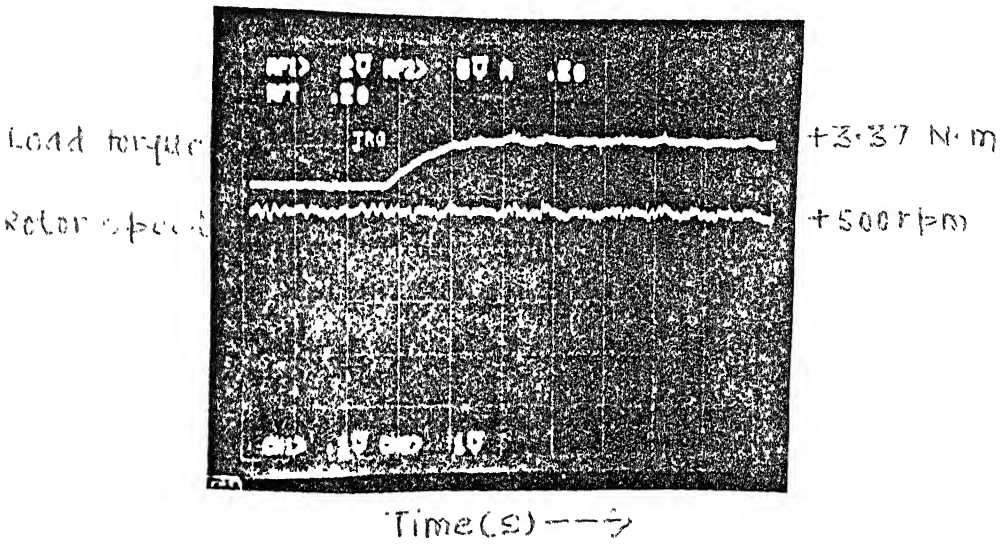
Fig. 4.4



(a)Rotor speed



(b)Load torque

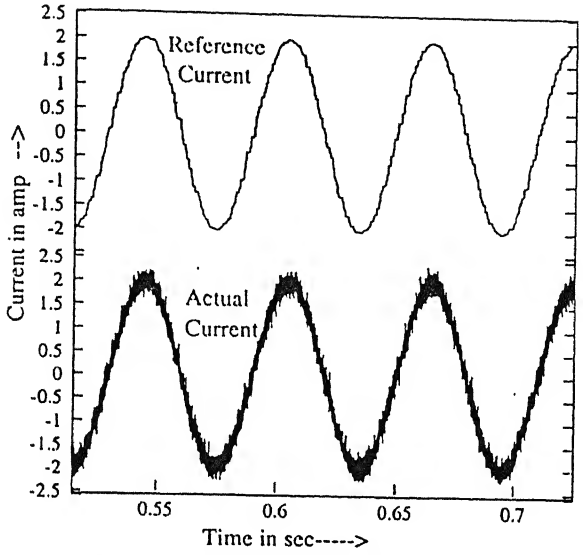


(c) upper waveform : Load torque(sensitivity 35N.m/V, scale .1V/div.)

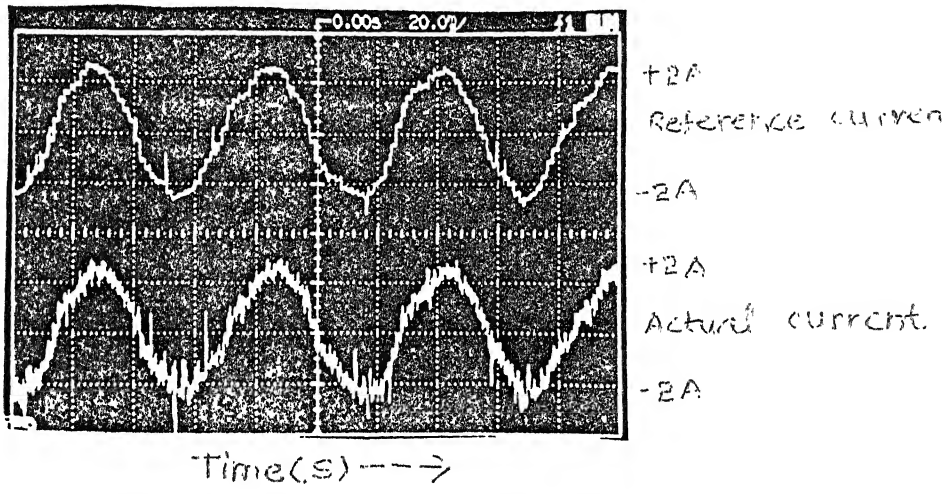
lower waveform : Rotor speed (scale 1V/div)

Waveforms: a,b Simulated
c experimental

Fig. 4.5



(a)phase 'a' reference and actual currents



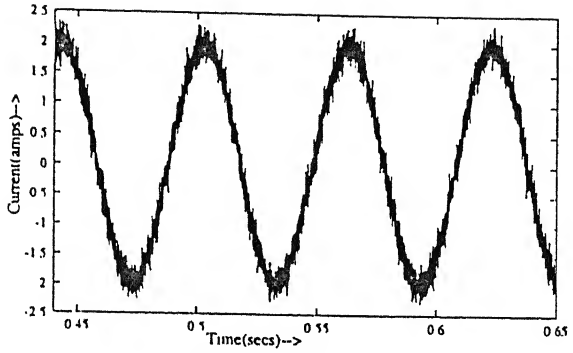
(b)phase 'a' reference and actual currents (sacle:1.0v/div)
sensitivity of current sensor 0.63v/amp

Reference and actual currents in phase 'a'

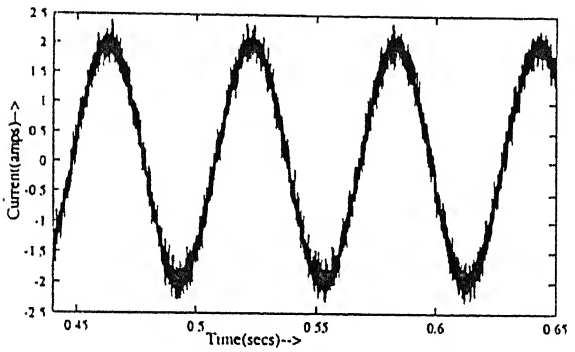
(a) Simulated

(b) Experimental

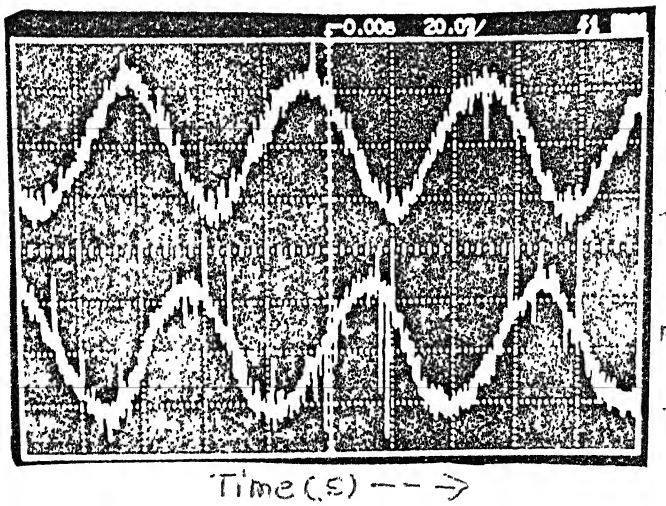
Fig. 4.6



(a)Phase 'a' current



(b)Phase 'b' current



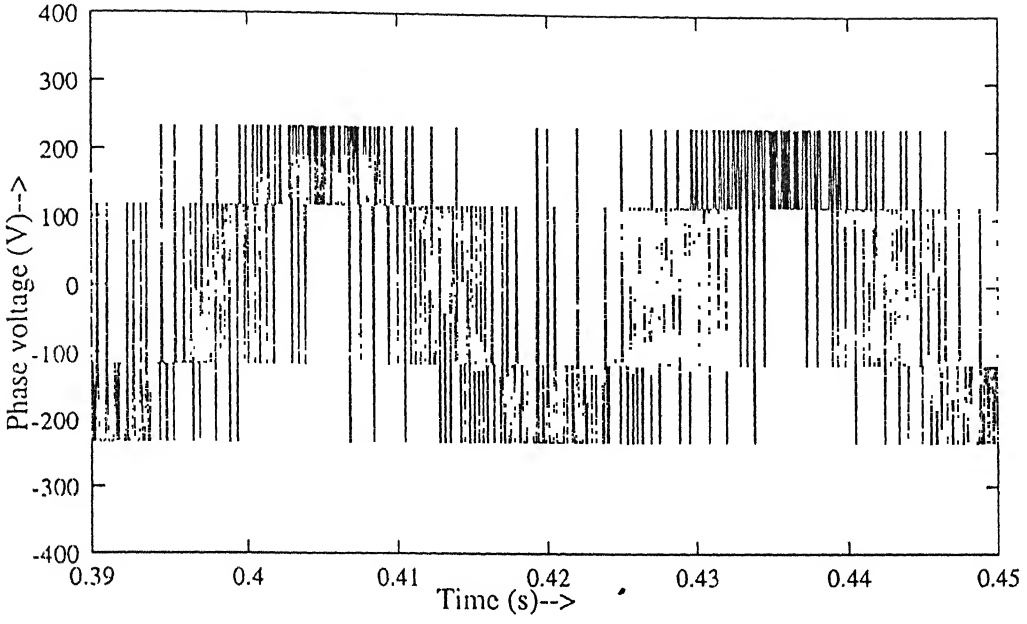
Scale 1 0V/div on vertical axis and Sensitivity of current sensor=.0 63v/amp

(c)Phae 'a' and phase 'b' currents

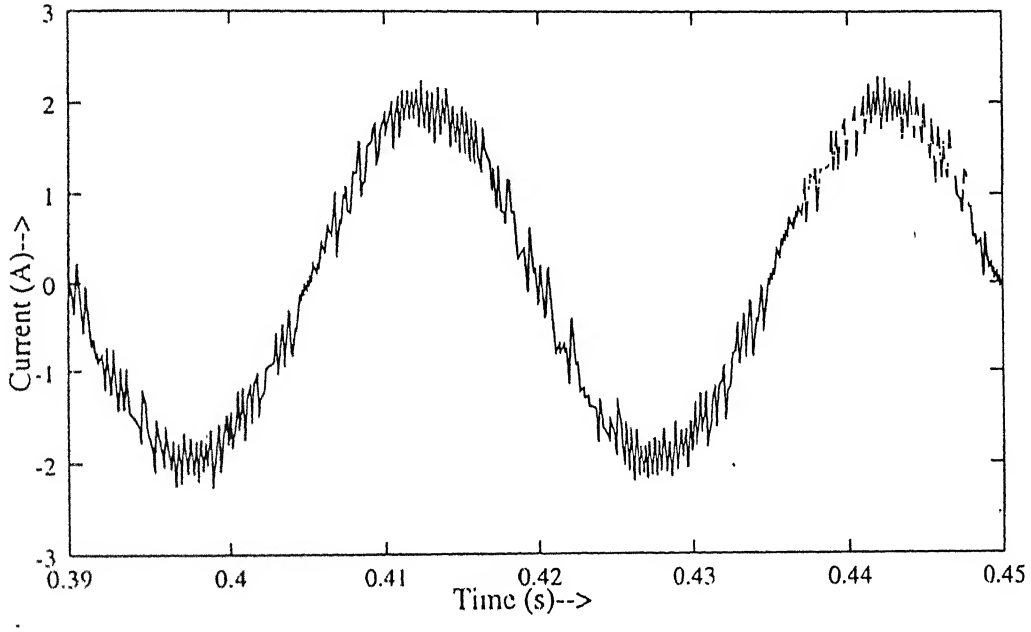
Phase 'a' and phase 'b' current waveforms: a,b Simulated

c Experimental

Fig. 4.7



(a) Phase voltage waveform



(b) Phase current waveform

Fig. 4.8 Simulated waveform of phase voltage and current of the induction motor

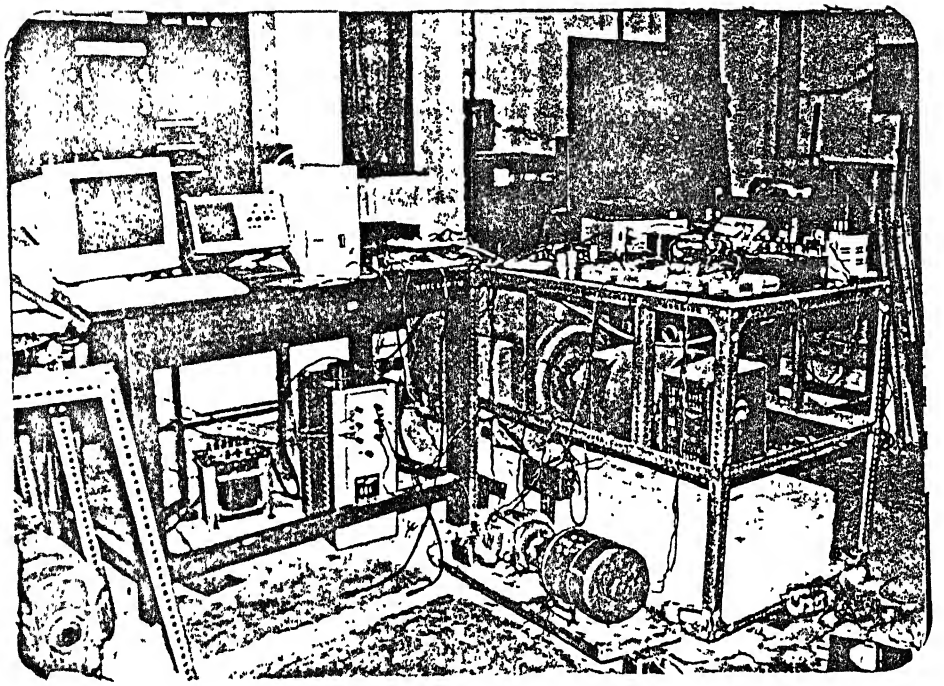


Fig. 4.9 A view of practical setup

Chapter 5

Conclusion

5.1 Salient features of the present work

Salient features of present thesis are

1. Design and development of a fuzzy logic controller for the closed loop speed control of vector controlled induction motor drive.
2. PC-based implementation of the vector controlled drive system.
3. Comparison of simulation and experimental results.

5.2 Scope for further research work

The proposed scheme combines advantages offered by fuzzy logic controller for controlling induction motor drive system without precise knowledge of system behavior. But very precise control of torque and speed can not be achieved since control rules defining behavior of the plant, are itself fuzzy. Fuzzy logic controller can give very fast torque response in transient state as demonstrated by practical and simulation results. So as discussed in [10], good features of fuzzy logic controller in transient state and excellent speed regulation of phase locked loop can be combined to improve over all accuracy of the plant.

In the proposed scheme the current references have been generated by calculating slip speed and adding it to actual rotor speed. The slip speed calculation is based on measuring rotor resistance and rotor inductance. Since resistance increases with increase in temperature, also with change in operating frequency the inductance changes. So with change in resistance and inductance, slip calculation no longer remains valid, this introduces error between actual slip speed and estimated slip speed. To account for variation in resistance due to temperature and frequency, calculation of resistance and inductance with varying operating conditions are

highly desirable. So fuzzy control with rotor parameter adaptation is desirable under varying parametric conditions.

In the proposed scheme synchronous speed is calculated by sensing actual machine speed in the form of tachometer output. Removal of speed sensor requires calculation of actual speed from known parameters such as voltage, currents and constants of induction motor. A closed loop sensor-less fuzzy vector controlled drive can be an area of further investigation.

References

1. F. Blaschke, "The principle of field orientation as applied to new transvector closed-loop control system for rotating field machines", *Siemens review, vol. 34, pp 22-220, may 1972.*

2. A Nabai, K. Otsuka, H. uchino, and R. Kurosawa, "An approach to flux control of induction motor operated with variable frequency power supply", *IEEE Trans. on Ind. Appl., Vol. IA-16, No.3, pp.342349,May/June 1980.*

3.E. Cerruto, A. Consoli, A. Raciti and A. Testa, "Fuzzy adaptive vector control of induction motor drives", *IEEE Trans. on Power Electronics,Vol.12,No. 6, pp. 1028-1039,Nov. 1997.*

4.B. Heber, L. Xu, and Y. Tang, "Fuzzy logic enhanced speed control of an Indirect Field Oriented induction machine drive", *IEEE Trans. on Power Electronics, Vol.12, No.5, pp. 772-778, Sept. 1997.*

5. G. C.D. Sousa, B. K. Bose and J. G. Cleland, "Fuzzy logic based on line efficiency optimization control of an Indirect Vector controlled induction motor drive", *IEEE Trans. on Industrial Electronics, Vol.42, No.2, pp. 192-198,April 1995.*

6.K. Ohnishi, H.Suzuki, K. Miyachi, and M. Terashima, "Decoupling control of secondary flux and secondary current in induction motor drive with controlled voltage source and its comparison with volts/hertz control", *IEEE Trans. on Ind. Appl., Vol. IA-21, No.1, pp.241-246, Jan. /Feb. 1985.*

7.H. Agrawal, "Simulation and realization of vector controlled induction motor drive based on MCS-80C196KC micro controller", *Mtech thesis 1996, IIT Kanpur.*

8. P.C. Krause, O. wasynczuk , S.D. Sudhoff, "Analysis of electric machinery", *IEEE press, Newyork.*

9. T. J. Ross, "Fuzzy Logic With Engineering Applications", *Mc Hill publications, Newyork.*

10. M. F. Lai, M. Nakano, and G. C. Hsieh, "Application of fuzzy logic in the phase-locked loop speed control of induction motor drive", *IEEE Trans. on Industrial Electronics Vol.43, No.6, pp.630-639, Dec. 1996.*

11. J. Jiang and J. Holtz, "High dynamic speed sensor-less ac drive with on-line model parameter tuning for steady-state accuracy", *IEEE Tran. On Industrial electronics, vol. 44. No. 2, April 1997.*

12. "User's manual, PCL-208 High performance data acquisition card," *Dynalog (INDIA) Ltd.*

Appendix A

Specifications of the 3-phase, 50 Hz induction motor

Ratings:

Power = 1.1 kW

Voltage = 415 V line to line

Current = 2.6 A.

Speed = 1410 rpm

Moment of inertia of the motor-dc generator system = 0.01 kg m^2

Rotor parameters:

Resistance = 4.3 ohm

Leakage inductance = 0.026 H

Stator parameters:

Resistance = 8.2 ohm

Leakage inductance = 0.026 H

Magnetizing inductance = 0.377 H

Specifications of the Tachogenerator:

Calibration factor = 6 V/ 1000 rpm.

Resistance = 20 ohms.

Max rpm = 2000.

Specifications of dc-shunt generator coupled with the IM

Ratings:

Power = 1 hp

Voltage = 230 V.

Current = 4.5 A.

Speed = 1500 rpm

IGBT specifications:

Voltage rating = 1200 V

Current rating = 50 A

Lock out delay time = 20 μ s.

Current sensor specifications

The LEM module LA 55-P is a current transducer for electronic measurement of currents with galvanic isolation between the primary (high power) and secondary (low power) circuits.

sensitivity = .63 V/A

Parameters for simulation

DC link voltage	350 V
Hysterisis band gap	±0.01 A
Stator flux component of current	2.0A
Step size	50 μ s
Sampling time for controller	500 μ s
PI controller gains	$K_p = 5$
	$K_i = 50$

Parameters for hardware implementation

DC link voltage	350 V
Hysterisis band gap	±0.01 A
Stator flux component of current	2.0 A
Step size	90 μ s
Sampling time for Controller	900 μ s

Appendix B

Listing of simulation program

```
#include<stdio.h>
#include<math.h>
#define PI 3.14159

/* defining linguistic variables */

#define  NLS                e<=-150
#define  NMS  e>=-200 &&  e<=-40
#define  NSS  e>=-50  &&  e<=0
#define  ZS   e>=-10  &&  e<=10
#define  PSS  e>=0    &&  e<=50
#define  PMS  e>=40   &&  e<=200
#define  PLS  e>=150
#define  NLC                de<=-30
#define  NSC  de>=-40 &&  de<=0
#define  ZC   de>=-5  &&  de<=5
#define  PSC  de>=0   &&  de<=40
#define  PLC  de>=30

/* Function for defuzzification */

float status(float e,float de);
float z,u,v,x[30],er,ch,mship,a[30];
float error(float,int);
float change(float,int);
float min(float,float);
int 1,j1,k1;
for(1=0;1<27;++1)
{
a[1]=0;
x[1]=0;
}
u=0;
v=0;
/* RULE 1 */
if(NLS)                          /*PL*/
{
j1=1;
```

```
mship=error(e,j1);
a[0]=.75*mship*(2-mship);
x[0]=3.5;
}
/* RULE 2,3,4,5,6 */
if(NMS)
{j1=2;
if(NLC)                          /*PL*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[1]=.75*mship*(2-mship);
x[1]=3.5;
}
if(NSC)                          /*PL*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[2]=.75*mship*(2-mship);
x[2]=3.5;
}
if(ZC)                           /*PS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[3]=1.5*mship*(2-mship);
x[3]=1.5;
}
if(PSC)                          /*PS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[4]=1.5*mship*(2-mship);
x[4]=1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[5]=1.5*mship*(2-mship);
x[5]=-1.5;
}
```

```
}
/* RULE 7 8 9 10 11 */
if(NSS)
{j1=3;
if(NLC)                          /*PL*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[6]=.75*mship*(2-mship);
x[6]=3.5;
}
if(NSC)                          /*PS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[7]=1.5*mship*(2-mship);
x[7]=1.5;
}
if(ZC)                           /*PS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[8]=1.5*mship*(2-mship);
x[8]=1.5;
}
if(PSC)                          /*PS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[9]=1.5*mship*(2-mship);
x[9]=1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[10]=1.5*mship*(2-mship);
x[10]=-1.5;
}

}
/* RULE 12 13 14 15 16 */
```

```
if(ZS)
{j1=4;
if(NLC)                          /*PS*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[11]=1.5*mship*(2-mship);
x[11]=1.5;
}
if(NSC)                          /*PS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[12]=1.5*mship*(2-mship);
x[12]=1.5;
}
if(ZC)                           /*Z*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[13]=.5*mship*(2-mship);
x[13]=0;
}
if(PSC)                          /*NS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[14]=1.5*mship*(2-mship);
x[14]=-1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[15]=1.5*mship*(2-mship);
x[15]=-1.5;
}
}
/* RULE 17,18,19,20,21 */
if(PSS)
{j1=5;
if(NLC)                          /*PS*/
```

```
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[16]=1.5*mship*(2-mship);
x[16]=1.5;
}
if(NSC)                          /*NS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[17]=1.5*mship*(2-mship);
x[17]=-1.5;
}
if(ZC)                           /*NS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[18]=1.5*mship*(2-mship);
x[18]=-1.5;
}
if(PSC)                          /*NS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[19]=1.5*mship*(2-mship);
x[19]=-1.5;
}
if(PLC)                          /*NL*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[20]=.75*mship*(2-mship);
x[20]=-3.5;
}
}
/* RULE 22,23,24,25,26 */
if(PMS)
{j1=6;
if(NLC)                          /*PS*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
```

```
mship=min(er,ch);
a[21]=1.5*mship*(2-mship);
x[21]=1.5;
}
if(NSC)                          /*NS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[22]=1.5*mship*(2-mship);
x[22]=-1.5;
}
if(ZC)                          /*NS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[23]=1.5*mship*(2-mship);
x[23]=-1.5;
}
if(PSC)                          /*NL*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[24]=.75*mship*(2-mship);
x[24]=-3.5;
}
if(PLC)                          /*NL*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[25]=.75*mship*(2-mship);
x[25]=-3.5;
}
}
/* RULE 27 */
if(PLS)                         /* NL*/
{j1=7;
mship=error(e,j1);
a[26]=.75*mship*(2-mship);
x[26]=-3.5;
}
for(l=0;l<27;++l)
{
u+=a[l]*x[l];
```

```
v+=a[l];
a[l]=0;
x[l]=0;
}
z=u/v;
return(z);
}
/* FUNCTION FOR SPEED ERROR */

float error(float e,int j1)
{
float g;
/* NEGATIVE LARGE */
switch(j1)
{
case 1:
if(NLS)
{
if(e>=-300)g=-(1+e/150);
else g=1;
}
break;
/* NEGATIVE MEDIUM */
case 2:
if(NMS)
{
if(e<=-120)g=(e+200)/80;
else g=-(e+40)/80;
}
break;
/* NEGATIVE SMALL */
case 3:
if(NSS)
{
if(e<=-25)g=(e+50)/25;
else g=-e/25;
}
break;
/* ZERO */
case 4:
if(ZS)
{
if(e<=0)g=(e+10)/10;
else
g=(10-e)/10;
}
break;
```

```
/* POSITIVE SMALL */
case 5:
if(PSS)
{
if(e<=25)g=e/25;
else g=(50-e)/25;
}
break;
/* POSITIVE MEDIUM */
case 6:
if(PMS)
{
if(e<=120)g=(e-40)/80;
else g=(200-e)/80;
}
break;
/* POSITIVE LARGE */
case 7:
if(PLS)
{
if(e<=300)g=e/150-1;
}
break;
}
return(g);

}

/* FUNCTION FOR ERROR CHANGE */

float change(float de,int k1)
{
float k;
switch(k1)
{
/* NEGATIVE LARGE */
case 1:
if(NLC)
{
if(de>=-60)k=-(de+30)/30;
else k=1;
}
break;
/* NEGATIVE SMALL */
case 2:
if(NSC)
{
```

```
if(de<=-20)k=(de+40)/20;
else k=-de/20;;
}
break;
/*ZERO*/
case 3:
if(ZC)
{
if(de<=0)k=(de+5)/5;
else
k=(5-de)/5;
}
break;
/* POSITIVE SMALL */
case 4:
if(PSC)
{
if(de<=20)k=de/20;
else
k=(40-de)/20;
}
break;
/* POSTIVE LARGE */
case 5:
if(PLC)
{
if(de<=60)k=(de-30)/30;
else k=1;
}
break;
}
return(k);
}
 /* FUNCTION FOR MINIMUM MEMBERSHIP */

 float min(float me,float mde)
 {
 if(me>mde)return(me);
 else return(mde);
 }
main()
{
int count=-1,lo=0,list,po=4;
float
e1=0,e0,wrs,iqss=0,idss,Kp,Ki,THs1=0,iqsm,THs2,THr=0,THe,h;
float sir, Vdc, Im, ias, ibs, ics;
float iqso,e,de,t0=0,T,m[5],n[5],o[5],p[5],g, ia, ib, ic;
```

```
float M=.357,Llr=.026,Lls=.026,d,f=60.,t=0,va,vb,vc;
float wrso,vds,vqs,iqs,iqr,ids,idr,J=.012;
float
rs=8.25,r=4.32,_t,siqr=0,siqs=0,sidr=0,sids=0,Te,Tl=0,wr=0,B=0
;
float
siqr1=0,siqs1=0,sidr1=0,sids1=0,iqs1=0,iqr1=0,ids1=0,idr1=0;
float vn, vao, vbo, vco, z;
char ch;
FILE *fp ,*fn ,*fm;
Fp = fopen("r" ,"w");
Fm = fopen("t", "w");
Fn = fopen("s", "r");
printf("Enter the controller you want to choose Fuzzy or PI P
,F\n");
ch=toupper(getchar());
printf("Enter the value  h vdc kp ki idss iqsm T wrs");
fscanf( fn, "%f %f %f %f %f %f %f %f %f\n",&h,&_t, &Vdc, &Kp,
&Ki, &idss, &iqsm, &T, &wrs);
g=M*(Llr+Lls)+Llr*Lls;
wrso=wrs;
do
{++count;
if(t<.01)wrs=0;
else if(t>=.01)wrs=wrso;
if(t>.3)wrs=-104.72;
if(count==10){count=0;
e1=-wrs+wr;
e=e1;
de=(e1-e0);
switch(ch)
{
case 'P':
iqss+=Kp*(-e1+e0)-Ki*e0*T;
break;
case'F':
iqss+=.07*status(10*e,1000*de);
break;
}
e0=e1;
if(iqss>iqsm)iqss=iqsm;
if(iqss<-iqsm)iqss=-iqsm;
THs1+=(iqss/idss)*(r/(Llr+M))*T;
THs2=atan2(iqss,idss);
THr+=wr*T;
THe=THs1+THs2+THr;
Im=sqrt(iqss*iqss+idss*idss);
```

```
/* calculating current references */

ias=Im*cos(THe);
ibs=Im*cos(THe-2*PI/3);
ics=Im*cos(THe+2*PI/3);
}

/* bang-bang logic */

if(ia<(ias-h))vao=Vdc;
if(ia>(ias+h))vao=0;
if(ib<(ibs-h))vbo=Vdc;
if(ib>(ibs+h))vbo=0;
if(ic<(ics-h))vco=Vdc;
if(ic>(ics+h))vco=0;
vn=(vao+vbo+vco)/3.;
va=vao-vn;
vb=vbo-vn;
vc=vco-vn;
vds=(va-.5*(vb+vc))*2./3;
vqs=(.866*(vb-vc))*2./3;
/* Runge-Kutta fourth order method */
for(list=0;list<4;++list)
{
iqs=((Llr+M)*siqs-M*siqr)/g;
iqr=((Lls+M)*siqr-M*siqs)/g;
ids=((Llr+M)*sids-M*sidr)/g;
idr=((Lls+M)*sidr-M*sids)/g;

m[list]=(vqs-rs*iqs)*_t;
n[list]=(vds-rs*ids)*_t;
o[list]=(-r*iqr+wr*sidr)*_t;
p[list]=(-r*idr-wr*siqr)*_t;
if(list!=3)
{
siqs+=m[list];
sids+=n[list];
siqr+=o[list];
sidr+=p[list];
}
}
iqs1=((Llr+M)*siqs1-M*siqr1)/g;
iqr1=((Lls+M)*siqr1-M*siqs1)/g;
ids1=((Llr+M)*sids1-M*sidr1)/g;
idr1=((Lls+M)*sidr1-M*sids1)/g;
siqs1+=(m[0]+2*m[1]+2*m[2]+m[3])/6.;
```

```
sids1+=(n[0]+2*n[1]+2*n[2]+n[3])/6.;
siqr1+=(o[0]+2*o[1]+2*o[2]+o[3])/6.;
sidr1+=(p[0]+2*p[1]+2*p[2]+p[3])/6.;
Te=(3/2.)*(po/2)*(sids1*iqs1-siqs1*ids1);
wr+=(po/2)*(1/J)*(Te-Tl-B*(2./po)*wr)*_t;
ia=ids1;
ib=-.5*ids1+.866*iqs1;
ic=-.5*ids1-.866*iqs1;
t+=_t;
sir=sqrt(sidr1*sidr1+siqr1*siqr1);
/* Output */
fprintf(fp,"%.5f %.4f %.4f %.4f %.4f %.4f %.4f %.4f
%.4f\n",t,Tl,sidr,siqr,sir,(60/(po*PI))*wr,Te,ia,ias+h);
siqs=siqs1;
sids=sids1;
siqr=siqr1;
sidr=sidr1;
}while(t<.6);
}
```

Appendix C

Listing of control program

```
#include<stdio.h>
#include<conio.h>
#include<dos.h>
#include<time.h>
#include<math.h>
#define PI 4.0*atan(1.0)

/* defining linguistic labels */

#define  NLS              e<=-150
#define  NMS  e>=-200 && e<=-40
#define  NSS  e>=-50  && e<=0
#define  ZS   e>=-10  && e<=10
#define  PSS  e>=0    && e<=50
#define  PMS  e>=40   && e<=200
#define  PLS  e>=150
#define  NLC              de<=-30
#define  NSC  de>=-40 && de<=0
#define  ZC   de>=-5  && de<=5
#define  PSC  de>=0   && de<=40
#define  PLC  de>=30
double ain0,ain1,ain2,ain3,ain4;
int port=0x300;

/* function for defuzzification */

float status(float e,float de)
{
float z,v,a[30],er,ch,mship,x[30];
float error(float,int),u=0;
float change(float,int);
float min(float,float);
int 1,j1,k1;
for(1=0;1<27;++1)
{
a[1]=0;
x[1]=0;
}
u=0;
```

```
v=0;
/* RULE 1 */
if(NLS)                          /*PL*/
{
j1=1;
mship=error(e,j1);
a[0]=.75*mship*(2 mship);
x[0]=3.5;
}
/* RULE 2,3,4,5,6 */
if(NMS)
{j1=2;
if(NLC)                          /*PL*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[1]=.75*mship*(2-mship);
x[1]=3.5;
}
if(NSC)                          /*PL*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[2]=.75*mship*(2-mship);
x[2]=3.5;
}
if(ZC)                           /*PS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[3]=1.5*mship*(2-mship);
x[3]=1.5;
}
if(PSC)                          /*PS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[4]=1.5*mship*(2-mship);
x[4]=1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
```

```
ch=change(de,k1);
mship=min(er,ch);
a[5]=1.5*mship*(2-mship);
x[5]=-1.5;
}
}
/* RULES 7,8,9,10, 11 */
if(NSS)
{j1=3;
if(NLC)                          /*PL*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[6]=.75*mship*(2-mship);
x[6]=3.5;
}
if(NSC)                          /*PS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[7]=1.5*mship*(2-mship);
x[7]=1.5;
}
if(ZC)                           /*PS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[8]=1.5*mship*(2-mship);
x[8]=1.5;
}
if(PSC)                          /*PS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[9]=1.5*mship*(2-mship);
x[9]=1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[10]=1.5*mship*(2-mship);
```

```
x[10]=-1.5;
}

}
/* RULES 12, 13, 14, 15, 16 */
if(ZS)
{j1=4;
if(NLC)                          /*PS*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[11]=1.5*mship*(2-mship);
x[11]=1.5;
}
if(NSC)                          /*PS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[12]=1.5*mship*(2-mship);
x[12]=1.5;
}
if(ZC)                           /*Z*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[13]=.5*mship*(2-mship);
x[13]=0;
}
if(PSC)                          /*NS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[14]=1.5*mship*(2-mship);
x[14]=-1.5;
}
if(PLC)                          /*NS*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);.
a[15]=1.5*mship*(2-mship);
x[15]=-1.5;
}
```

```
}
/* RULES 17, 18, 19, 20, 21 */
if(PSS)
{j1=5;
if(NLC)                          /*PS*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[16]=1.5*mship*(2-mship);
x[16]=1.5;
}
if(NSC)                          /*NS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[17]=1.5*mship*(2-mship);
x[17]=-1.5;
}
if(ZC)                           /*NS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[18]=1.5*mship*(2-mship);
x[18]=-1.5;
}
if(PSC)                         /*NS*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[19]=1.5*mship*(2-mship);
x[19]=-1.5;
}
if(PLC)                        /*NL*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[20]=.75*mship*(2-mship);
x[20]=-3.5;
}
}
/* RULES 22, 23, 24, 25, 26, */
if(PMS)
```

```
{j1=6;
if(NLC)                          /*PS*/
{k1=1;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[21]=1.5*mship*(2-mship);
x[21]=1.5;
}
if(NSC)                           /*NS*/
{k1=2;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[22]=1.5*mship*(2-mship);
x[22]=-1.5;
}
if(ZC)                           /*NS*/
{k1=3;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[23]=1.5*mship*(2-mship);
x[23]=-1.5;
}
if(PSC)                          /*NL*/
{k1=4;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[24]=.75*mship*(2-mship);
x[24]=-3.5;
}
if(PLC)                          /*NL*/
{k1=5;
er=error(e,j1);
ch=change(de,k1);
mship=min(er,ch);
a[25]=.75*mship*(2-mship);
x[25]=-3.5;
}
}
/* RULE 27 */
if(PLS)                         /* NL*/
{j1=7;
mship=error(e,j1);
a[26]=.75*mship*(2-mship);
```

```
x[26]=-3.5;
}
for(l=0;l<27;++l)
{
u+=a[l]*x[l];
v+=a[l];
a[l]=0;
x[l]=0;
}
z=u/v;
return(z);
}
/* FUNCTION FOR SPEED ERROR */

float error(float e,int j1)
{
float g;
/* NEGATIVE LARGE */
switch(j1)
{
case 1:
if(NLS)
{
if(e>=-300)g=-(1+e/150);
else g=1;
}
break;
/* NEGATIVE MEDIUM */
case 2:
if(NMS)
{
if(e<=-120)g=(e+200)/80;
else g=-(e+40)/80;
}
break;
/* NEGATIVE SMALL */
case 3:
if(NSS)
{
if(e<=-25)g=(e+50)/25;
else g=-e/25;
}
break;
/*  ZERO  */
case 4:
if(ZS)
{
```

```
if(e<=0)g=(e+10)/10;
else
g=(10-e)/10;
}
break;
/* POSITIVE SMALL */
case 5:
if(PSS)
{
if(e<=25)g=e/25;
else g=(50-e)/25;
}
break;
/* POSITIVE MEDIUM */
case 6:
if(PMS)
{
if(e<=120)g=(e-40)/80;
else g=(200-e)/80;
}
break;
/* POSITIVE LARGE */
case 7:
if(PLS)
{
if(e<=300)g=e/150-1;
}
break;
}
return(g);

}

/* FUNCTION FOR ERROR CHANGE */

float change(float de,int k1)
{
float k;
switch(k1)
{
/* NEGATIVE LARGE */
case 1:
if(NLC)
{
if(de>=-60)k=-(de+30)/30;
else k=1;
}
```

```
break;
/* NEGATIVE SMALL */
case 2:
if(NSC)
{
if(de<=-20)k=(de+40)/20;
else k=de/20;;
}
break;
/* ZERO */
case 3:
if(ZC)
{
if(de<=0)k=(de+5)/5;
else
k=(5-de)/5;
}
break;
/* POSITIVE SMALL */
case 4:
if(PSC)
{
if(de<=20)k=de/20;
else
k=(40-de)/20;
}
break;
/* POSTIVE LARGE */
case 5:
if(PLC)
{
if(de<=60)k=(de-30)/30;
else k=1;
}
break;
}
return(k);
}
/* FUNCTION FOR MINIMUM MEMBERSHIP */

float min(float me,float mde)
{
if(me>mde)return(me);
else return(mde);
}
main()
{ clock_t start,end;
```

```
int sa=0,sb=0,sc=0;
int count=19;
float e1=0,e0=0,wrs=-
104.72,iqss=0,idss=2.0,Kp=5.0,Ki=50.0,THs1=0,iqsm=3.0,THs2=0,T
Hr=0,THe=0,h=0.001;
double Im=0,ias=0,ibs=0,ics=0,r=4.32;
double e=0,de=0,T=.001780,ia=0,ib=0,ic=0;
double M=.377,Llr=.026,d,t=0;
double _t=.000089,wr=0;
int ch1=0,ch2=0,ch3=0;
void fireout(int,int,int);
void csensor(void);
char ch;
FILE *fn;
fn=fopen("r","w");
printf("Enter the controller you want to choose Fuzzy or PI P
,F\n");
ch=toupper(getchar());
 start=clock();
do
{
++count;
csensor();
if(count==10){count=0;
wrs=32*ain4;
if(t>1.5)wrs=104.72;
wr=34.89*ain2;
e1=-wrs+wr;
e=e1;
de=e1-e0;

/* Fuzzy controller simulation */

switch(ch)
{
case 'P':
iqss+=.2*(Kp*(-e1+e0)-Ki*e0*T);
case 'F':
iqss+=0.07status(10*e,1000*de);
break;
}
e0=e1;
if(iqss>iqsm)iqss=iqsm;
if(iqss<-iqsm)iqss=-iqsm;
THs1+=(iqss/idss)*(r/(Llr+M))*T;
THs2=atan2(iqss,idss);
THr+=wr*T;
```

```
THe=THs1+THs2+THr;
Im=sqrt(iqss*iqss+idss*idss);
/* Setting current reference */
ias=Im*cos(THe);
ibs=Im*cos(THe-2*PI/3);
ics=Im*cos(THe+2*PI/3);
ch1=floor(4095*(1.5-(1.8/28.6)*ias)/5.0);
ch2=ch1/16;
ch3=16*(ch1-ch2*16);
outport(port+4,ch3);
outport(port+5,ch2);
}

ib=-1.5713*ain0;
ic=-1.5713*ain1;
ia=-1.5713*ain3;

/* bang-bang logic */
if(ia<(ias-h))sa=1;
if(ia>(ias+h))sa=0;
if(ib<(ibs-h))sb=1;
if(ib>(ibs+h))sb=0;
if(ic<(ics-h))sc=1;
if(ic>(ics+h))sc=0;
fireout(sa,sb,sc);
t+=_t;
}while(t<5.0);
end=clock();

printf("Time=%f  T=%f",(end-start)/CLK_TCK,T);

outport(port+3,0x00);/* Inhibiting firing pulses */

}

/* FUNCTION FOR SENDING FIRING PULSES TO OUT PORT
void fireout(int sa,int sb,int sc)
{
int mp;
mp=4*sc+2*sb+sa;
switch(mp){
case 0:
outport(port+3,0x00);
break;
case 1:
outport(port+3,0x01);
break;
```

```
case 2:
outport(port+3,0x02);
break;
case 3:
outport(port+3,0x03);
break;
case 4:
outport(port+3,0x04);
break;
case 5:
outport(port+3,0x05);
break;
case 6:
outport(port+3,0x06);
break;
case 7:
outport(port+3,0x07);
break;
}
return;
}
/* FUNCTION FOR SENSING CURRENT */
 void csensor(void)

     {
    int st,uni=1,s_end=1,diff,bip,start=0,stop=4;
    int dtl,dth,adl,adt,c_reg,s_ch_val,r_ch;
    int chv,pool=0;
    char number;
    int val;


/*  ******** STEP 1: INITIALIZE & SELECTE SOFTWARE TRIGGER
*************** */
/*  PORT+9 ---- CONTROL REGISTER
*/
    val=0x70;
    outportb (port+9,val);
    c_reg=inportb (port+9);
    if (c_reg != val)
     {
     printf ("pc1-718 hardware verification failed!\n");
     exit(0);
     }
    outportb (port+8,1);       /* CLEAR INTERRUPT REQUEST  */
```

```
/*  ******** STEP 3: SET SCAN CHANNEL RANGE
***************************** */

    s_ch_val=stop*16+start;      /* SET SCAN CHANNEL VALUE */
    outportb (port+2,s_ch_val);
    r_ch=inportb (port+2);       /* READ BACK CHANNEL VALUE */
    if (r_ch != s_ch_val)
    {
     printf ("set scan channel failed!\n");
     exit(0);
    }
/*  ******** STEP 4: PERFORM SINGLE A/D CONVERSION  **********
*/
do {
    outportb (port,0);
reread:    st=inport(port+8);
    if ((st & 0x80)==0x80)
    goto reread;
       dtl=inportb(port);
       dth=inportb(port+1);
       adl=dtl/16;
       adt=dth*16+adl;
       chv=dtl-adl*16;
/* Reading motor currents, reference speed and rotor speed */
     switch(chv)
     {case 0:
     ain0=(adt-2048)*5.0/2048;
     break;
     case 1:
     ain1=(adt-2048)*5.0/2048;
     break;
     case 2:
     ain2=(adt-2048)*5.0/2048;
     break;
     case 3:
     ain3=(adt-2048)*5.0/2048;
     break;
     case 4:
     ain4=(adt-2048)*5.0/2048;
     break;
     }
     ++pool;
    } while(pool<5);
}
```

Date Slip.

This book is to be returned on the date last stamped.